Universidad Zaragoza
1542

Trabajo Fin de Grado

Seguimiento de la señal de Galileo E1 OS para UAVs

Tracking of the Galileo E1 OS Signal for UAVs

Autor

Mario Miravete del Castillo

Directores

Jakob Jakobsen
Daniel Haugård Olesen

Escuela de Ingeniería y Arquitectura

2017

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. Mario Miravete del Castillo,

con nº de DNI 25197888P en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Grado, (Título del Trabajo) Seguimiento de la señal de Galileo E1 OS para UAVs (título en inglés: Tracking of the Galileo E1 OS Signal for UAVs),

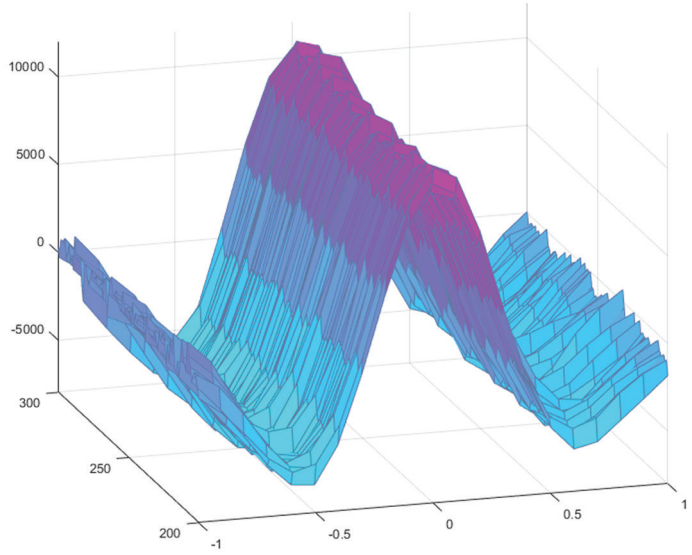
es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 6 de Julio de 2017

Fdo: Mario Miravete del Castillo

Tracking of the Galileo E1 OS Signal for UAVs

Bachelor's Thesis



Mario Miravete del Castillo
June 2017

Supervisors:
Jakob Jakobsen
Daniel Haugård Olesen

DTU Space
National Space Institute

DTU Space
National Space Institute
Technical University of Denmark

Elektrovej, building 327
DK - 2800 Kgs. Lyngby
Tel (+45) 4525 9500
Fax (+45) 4525 9575

www.space.dtu.dk

Summary

This project is related to tracking of the Galileo E1 OS signal. The objective is to perform the baseband processing of Galileo signals using *Matlab*, thus the scope of the project is within digital processing of radio-frequency signals.

The implementation is based on an existing toolbox developed at DTU Space, Division of Geodesy, for GPS, which has been adapted to cope with the Galileo signal. Besides this, data has been collected using a software receiver and the toolbox has been tested. Furthermore, a multipath study has been carried out using a multi-correlator strategy.

The thesis is divided into four main blocks. The first one introduces the Galileo signal as well as some software receiver theory as a background for the implementation, which is described right afterwards. Then, the data collection set-up is shown and some results are plotted, along with a discussion. Finally, multipath is analyzed in a separated chapter consisting of a small theory section, the implementation modifications and the results section.

Resumen (Spanish)

Este proyecto está relacionado con el seguimiento de la señal E1 OS de Galileo. El objetivo es llevar a cabo el procesado en banda base de señales de Galileo usando *Matlab*, por lo que el alcance del proyecto está dentro del procesado digital de señales de radiofrecuencia.

La implementación se basa en un toolbox existente desarrollado en el departamenteo de DTU Space para GPS, que ha sido adaptado para aceptar la señal de Galileo. Además de esto, se han recogido datos usando un receptor software y el toolbox ha sido probado. Asimismo, se ha llevado a cabo un estudio de multipath usando una estrategia multi-correlador.

La tesis está dividida en cuatro bloques principales. El primero introduce la señal de Galileo, así como algo de teoría sobre receptores software como background para la implementación, que se describe justo después. Posteriormente, se muestra el setup para la recogida de datos junto con algunos resultados y la discusión de los mismos. Finalmente, se analiza el multipath en un capítulo separado, que consiste en una pequeña sección de teoría, las modificaciones en la implementación y la seccion de resultados.

Preface

This B.Sc.Eng. Thesis was prepared at the department of Space (National Space Institute) at the Technical University of Denmark in fulfillment of the requirements for acquiring a B.Sc.Eng degree in Telecommunication Technologies and Services at the University of Zaragoza (Spain).

I hereby certify that this thesis has been composed by me and is based on my own work, unless stated otherwise. No other person's work has been used without due acknowledgement in this thesis. All references and verbatim extracts have been quoted, and all sources of information, including graphs and data sets, have been specifically acknowledged.

Kongens Lyngby, July 6, 2017

Mario Miravete del Castillo

Contents

List of Figures

Acronyms

ACF Autocorrelation Function.

ADC Analog to Digital Converter.

BPSK Binary Phase Shift Keying.

CBOC Composite Binary Offset Carrier.

CDMA Code Division Multiple Access.

DLL Delay Lock Loop.

E-P-L Early Prompt Late.

ESA European Space Agency.

FPGA Field Programmable Gate Array.

GNSS Global Navigation Satellite System.

GPS Global Positioning System.

GSTB Galileo System Testbed.

GUI Graphical User Interface.

IF Intermediate Frequency.

ION Institute Of Navigation.

LOS Line Of Sight.

M-Cr Multi-Correlator.

NCO Numeric Controlled Oscillator.

OS Open Service.

PLL Phase Lock Loop.

PRN Pseudo Random Noise.

PRS Public Regulated Service.

RHCP Right Hand Circular Polarization.

SDR Software Defined Radio.

UAV Unmanned Aerial Vehicles.

UHD USRP Hardware Driver.

USB Universal Serial Bus.

USRP Universal Software Radio Peripheral.

Symbols

BLK_{size} Block size input for the tracking loop.

B_N Noise bandwidth.

C_{E1-B} PRN code for the B channel.

C_{E1-C} PRN code for the C channel.

C_{E1-C}^{prim} Primary PRN code for the C channel.

C_{E1-C}^{sec} Secondary PRN code for the C channel.

$C_{L1C/A}$ PRN code for the GPS L1 C/A signal.

Cos_C Cosine carrier signal.

D_{E1-B} Navigation data.

D_{L1} Navigation data for the GPS L1 band.

F_C Code frequency.

F_S Sampling frequency.

I In-phase signal/sample.

IE Early in-phase signal/sample.

IL Late in-phase signal/sample.

IP Prompt in-phase signal/sample.

L_C Code Length.

N_S Number of samples per code.

$Ph_{carrier}$ Carrier phase.

Ph_{code} Code phase.

Q Quadrature signal/sample.

QE Early quadrature signal/sample.

QL Late quadrature signal/sample.

QP Prompt quadrature signal/sample.

$SC_{BOC(1,1)}$ 1 MHz square subcarrier.

$SC_{BOC(6,1)}$ 6 MHz square subcarrier.

S_{E1} Satellite signal transmitted in the E1 band.

S_{L1} Satellite signal for the GPS L1 band.

Sin_C Sine carrier signal.

T_S Sampling period.

T_C Chip period.

α Scaling factor of the main subcarrier.

β Scaling factor of the second subcarrier.

ω Angular frequency of an arbitrary signal.

ϕ Phase of an arbitrary signal.

e_{E1-B} B channel signal.

e_{E1-C} C channel signal.

f_C Carrier frequency.

$p(t)$ Baseband pulse with the duration of one chip period.

t Absolute temperature in K.

CHAPTER 1

Introduction

This thesis is included within the framework of Software Defined Radio (SDR) systems. A wireless system consists of three subsystems: antenna, RF front-end and baseband processor. In the early days, all the processing was done in the analog domain, while nowadays analog components have been replaced by digital components [1].

Ideally, the process should continue in the digital domain (software) after the antenna and an A/D converter. Nevertheless, this is still not achievable mainly due to two reasons: the signal is too weak (the voltage is below thermal noise) and the frequency is too high for the converters to operate. Therefore, some kind of analog RF front-end is necessary to solve this problem and act as signal conditioner for the converter and posterior software baseband processing. This signal provided to the ADC is usually higher in voltage and lower in frequency (IF). This strategy is described in Figure 1.1 and is the structure used in this project.

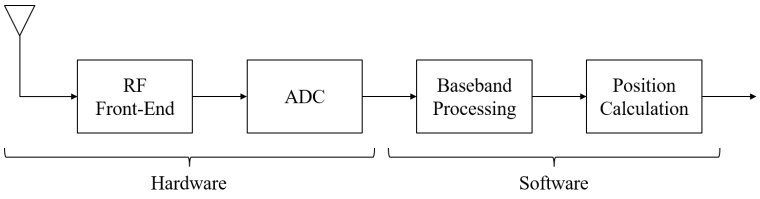


Figure 1.1: Software Defined Radio paradigm.

1.1 Problem description

The proposed problem is to achieve acquisition and tracking of Galileo satellites from a previously collected data set (IF samples). In other words, the objective is to perform the software baseband processing for a GNSS receiver focusing on the Galileo signal.

This is a well known problem and it has been treated before. Some approaches can easily be found in the literature, for example in [2], [3] or [4]. Due to this fact, an additional task is raised: a multipath mitigation solution for Galileo tracking and a small performance study.

According to the European Union Law:

> *"In short, Galileo would have to harness the potential for application of a satellite navigation system to civilian uses by striving to fill the gaps in the GPS and reinforcing the reliability of GNSS. It would have to provide global coverage immediately."*
> (Satellite navigation: Galileo (EU legislation, 1999) [5])

Almost 20 years after the project begun, the constellation is still incomplete but 18 satellites have been launched (13 operational) and the system is expected to be fully working within the next decade. Thus the importance of the project is not questionable.

1.2 Methodology and time plan

The project work has been divided in several tasks (see Figure 1.2), distributed along the duration of the project: 20 weeks from the 13^{th} of February 2017 to the 26^{th} of June 2017.

These main tasks are also the content of this thesis and are grouped in four main blocks: theoretical background, implementation, results/performance and multipath study. The implementation is developed in *Matlab*, since it is based on an existing toolbox for GPS applications developed at DTU Space [6].

Figure 1.2 shows the time plan that has been used for this project, were it can be noticed the weight of each part in relation to dedicated time.

Task	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13	W14	W15	W16	W17	W18	W19	W20
	13/02	20/02	27/02	06/03	13/03	20/03	27/03	03/04	10/04	17/04	24/04	01/05	08/05	15/05	22/05	29/05	05/06	12/06	19/06	26/06
Literature reading																				
Toolbox study																				
Implementation																				
Acquisition																				
Tracking																				
Data collection																				
Multipath research																				
Literature reading																				
Implementation																				
Data collection																				
Testing																				
Thesis writting																				
Hand-in																				
Defense																				

Figure 1.2: Thesis time plan (schedule).

CHAPTER 2
Theoretical Background

In order to understand how the implemented system works, it is essential to have some knowledge about satellite positioning systems and the Galileo signal in particular, as well as SDR architectures and operations. This chapter introduces the basic concepts regarding these topics.

All the information about the signal is obtained from the official Galileo "Signal In Space Interface Control Document" [7], where the reader can get a deeper understanding about everything related to this. For further thoughts on receiver architectures and operations, the reader is referred to [2], [8], [9], [10].

2.1 Satellite positioning systems

According to [11], "Satellite-based positioning is the determination of positions of observing sites on land or at sea, in the air and in space by means of artificial satellites". A constellation of satellites constantly transmitting data (broadcast) make it possible for the user to determine the position in space. Therefore, three satellites are needed for every coordinate component and a fourth to calculate the clock drift.

Galileo space segment is composed of a constellation of 30 satellites with three orbital planes at 56° inclination. Each plane contains 8 operational satellites spaced 45°. The orbit altitude is 23222 km, which means that the space loss is extremely high: the guaranteed signal power is -157 dBW, which is actually below thermal noise.

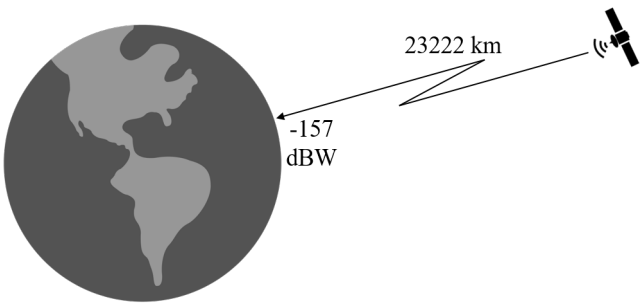


Figure 2.1: Galileo satellite squeme.

Let the Boltzmann's constant be $k = 1.38 \cdot 10^{-23}$ $[J/K]$, the absolute temperature t $[K]$ and the noise bandwidth B_N $[Hz]$, for a temperature of 25 degrees and a bandwidth of 4 MHz (see Figure 2.9), the noise power is:

$$\begin{aligned} P_N &= 10\log(ktB_N) \\ &= 10\log(298 \cdot 4 \cdot 10^6 \cdot 1.38 \cdot 10^{-23}) \\ &= -137.83\ dBW \quad (-107.83\ dBm) \end{aligned} \tag{2.1}$$

The *free space loss* between the satellite and the receiver illustrated in Figure 2.1 can be calculated easily as:

$$FSL = 20\log\left(\frac{4\pi df}{c}\right) \tag{2.2}$$

Being d the distance to the satellite (23222 km), f the carrier frecuency (between 1 and 2 GHz for Galileo) and c the speed of light ($3 \cdot 10^8$ m/s).

As all satellites are sharing the same carrier-frequency, a solution to achieve reception from different satellites is necessary: Code Division Multiple Access (CDMA) is the strategy adopted, which is a direct-sequence spread-spectrum technique [12]. This means that different spreading codes per signal, per frequency, and per satellite are used.

Figure 2.2 illustrates the phenomenon, basically consisting of the signal being multiplied by a spreading code (with a higher chip rate than the signal data rate). This spread signal is transmitted over the air and received by the receiving antenna. Note that all the spread signals from all the satellites are present at the receiver. When multiplied again by he same spreading code and filtered, the original signal is recovered.

On top of this, narrowband interference to the receiver is reduced, since it is also multiplied by the reception replica of the code, being spread in frequency and distributing the energy over non-interest bands.

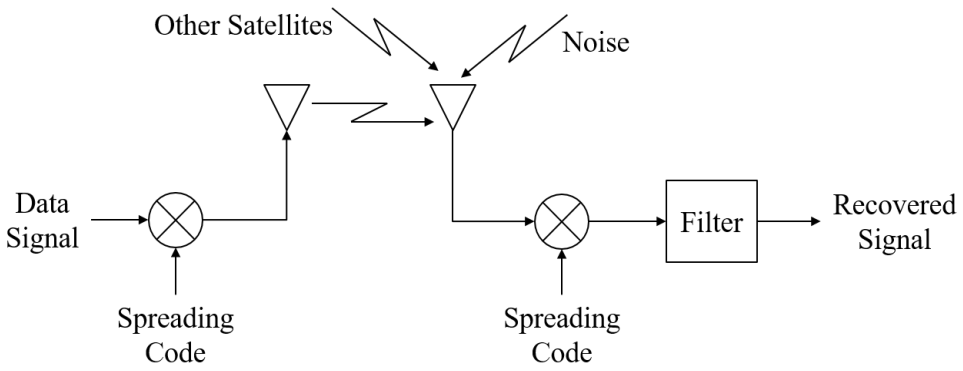
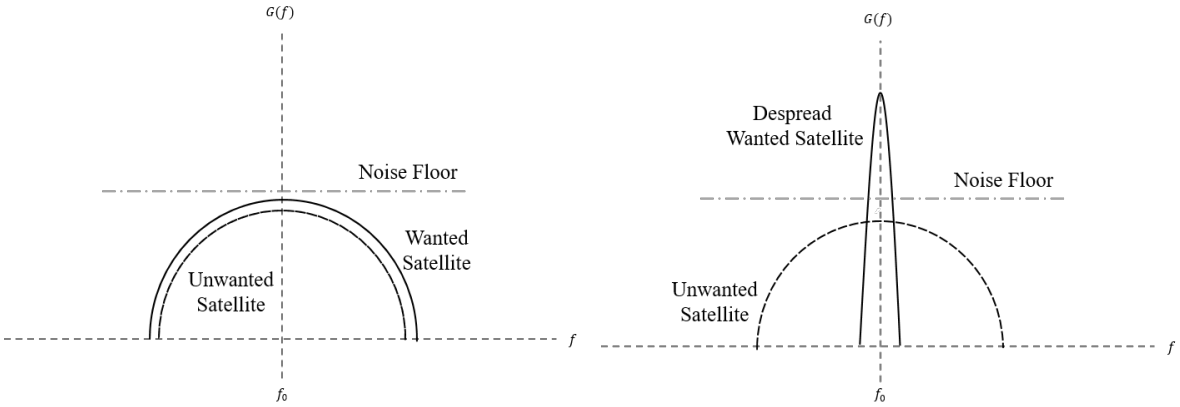


Figure 2.2: CDMA technique.

The process is shown in Figure 2.3 in the frequency domain. The consequence of multiplying by a pseudo-random code sequence is the spreading of the signal over frequency, which in the Galileo case is below the thermal noise floor. When despread, the signal has a smaller bandwidth and high power level to be treated.



(a) Spread spectrum reception (b) Output of the correlation

Figure 2.3: Spread spectrum reception (a) and output of the correlation (b).

2.2 Galileo signal plan

As stated in the Galileo Interface Document: "Three independent CDMA signals, named E5, E6 and E1, are permanently transmitted by all Galileo satellites." Figure 2.4 illustrates the radio-frequency air interface according to this.

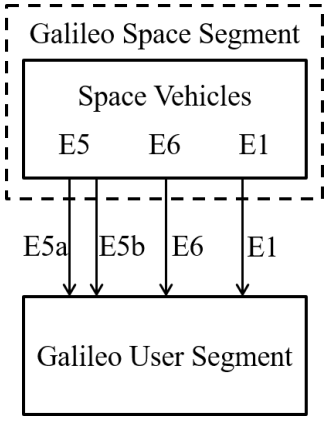


Figure 2.4: Galileo radio frequency air interface.

Section 2.2.1 describes the Galileo frequency bands more in detail.

2.2.1 Frequency bands

Figure 2.5 shows a frequency plot in the L band indicating both the Galileo and Global Positioning System (GPS) operating bands. Carrier frequencies are specified in Table 2.1 and receiver reference bandwidths in Table 2.2.

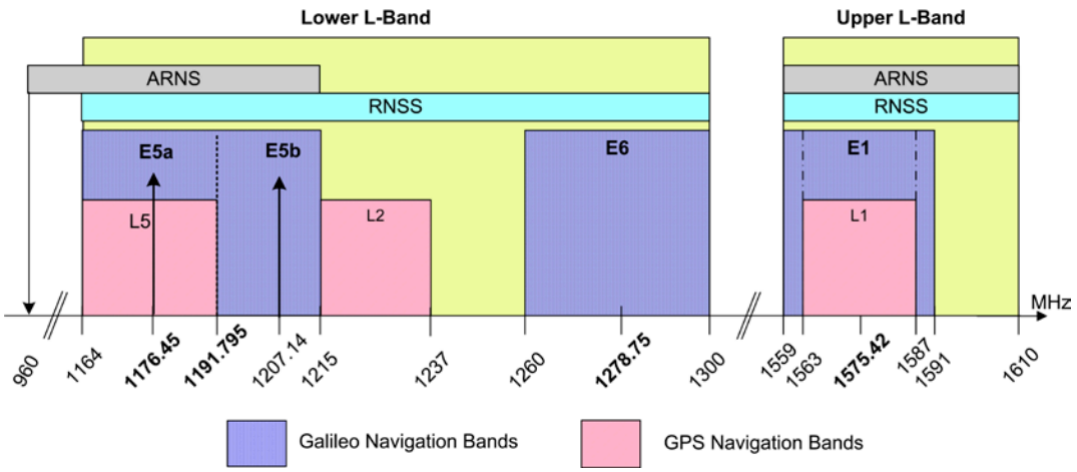


Figure 2.5: Galileo frequency plan. Source: Galileo Interface Document.

Table 2.1: Carrier frequency per signal (according to Galileo frequency plan).

Signal	Carrier Frequency (MHz)
E1	1575.420
E6	1278.750
E5	1191.795
E5a	1176.450
E5b	1207.140

Table 2.2: Galileo signals receiver reference bandwidths.

Signal	Bandwidth (MHz)
E1	24.552
E6	40.920
E5	51.150
E5a	20.460
E5b	20.460

In this project the focus is on the E1 signal, and further sections will explain the main characteristics as an overview.

2.2.2 E1 OS signal characteristics

E1 Galileo signal consists of three multiplexed components: the E1 Open Service (OS) data channel (E1-B), the E1 OS pilot channel (E1-C) and the E1 PRS channel (E1-A). OS and PRS are modulated in the in-phase and quadra-phase components respectively.

This project is focusing on the Open Service signal (In-phase component of the E1 signal). Further details about this signal are explained in the following sections.

2.2.2.1 Modulation

The modulation used by Galileo E1 OS is Composite Binary Offset Carrier (CBOC). The basic idea behind this modulation scheme is that the Binary Phase Shift Keying (BPSK) signal is then multiplied by a linear combination of two square subcarriers of equal or higher rate than the code rate of the signal. The whole process is shown in Figure 2.6 from the navigation message to the final signal sent by the satellites.

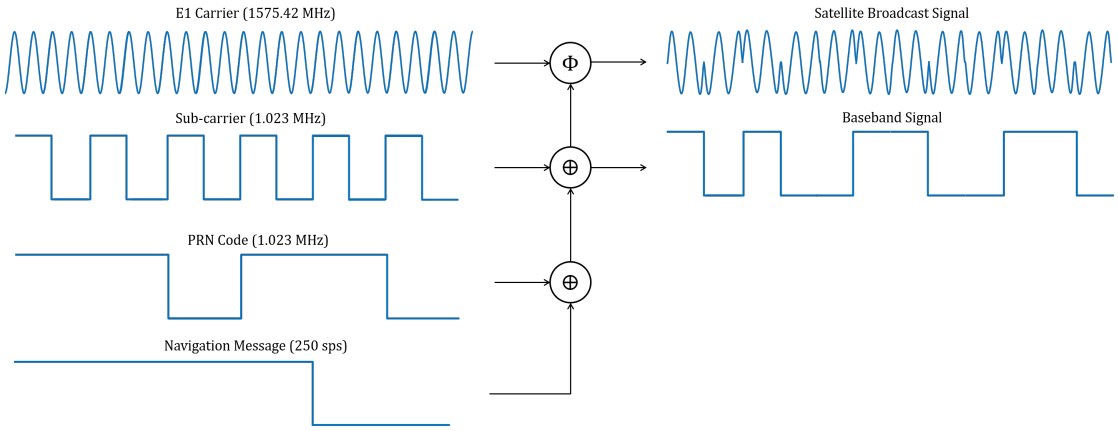


Figure 2.6: CBOC generation process.

First the signal is multiplied by the spreading code, then by this linear combination of sub-carriers and afterwards multiplied by the E1 carrier.

In the frequency domain, this means that the main peak of the BPSK signal is split in two side peaks at the subcarrier lower rate. Additional peaks appear at the subcarrier higher rate too.

Furthermore, regarding the ACF, the traditional BPSK triangle shape is modified by a three-peak shape, which is an improvement in relation to multipath mitigation (see Chapter 5). This is illustrated in Figure 2.7.

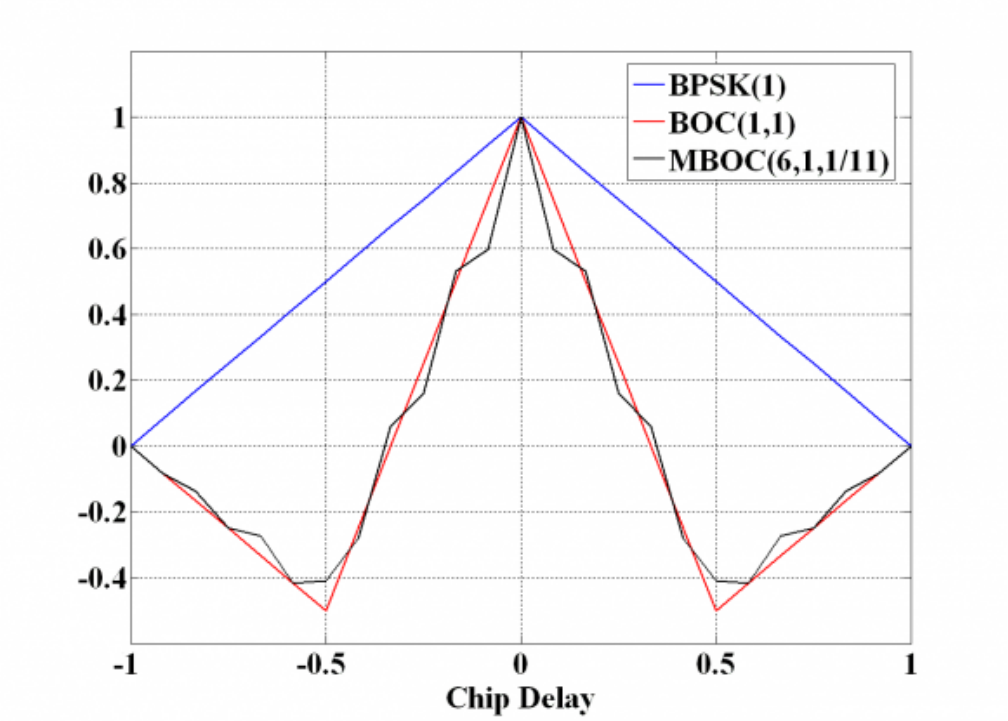


Figure 2.7: ACF of BPSK(1), BOC(1,1) and MBOC(6,1,1/11). Source: Navipedia.

The CBOC used by Galileo E1 OS signals is BOC(N,M,k) being N the scaling factor of the higher rate subcarrier, M the scaling factor of the second subcarrier and k the power fraction of the second subcarrier, being *1-k* the power of the first subcarrier.

BOC (6,1,1/11) is the chosen modulation scheme for the E1 OS signal. According to this, the subcarrier combinations are represented in Figure 2.8

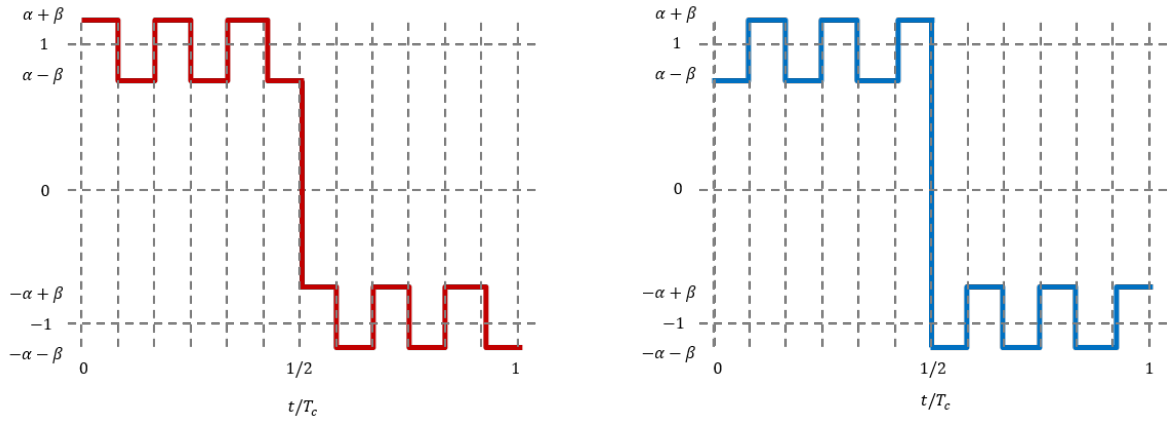


Figure 2.8: One period of the CBOC subcarrier. E1-B (red) and E1-C (blue).

This 6 MHz and 1 MHz subcarriers mean that the original spectrum will be split in two peaks at the carrier frequency ± 1 MHz and two peaks will appear at the carrier frequency ± 6 MHz. This is shown in Figure 2.9.

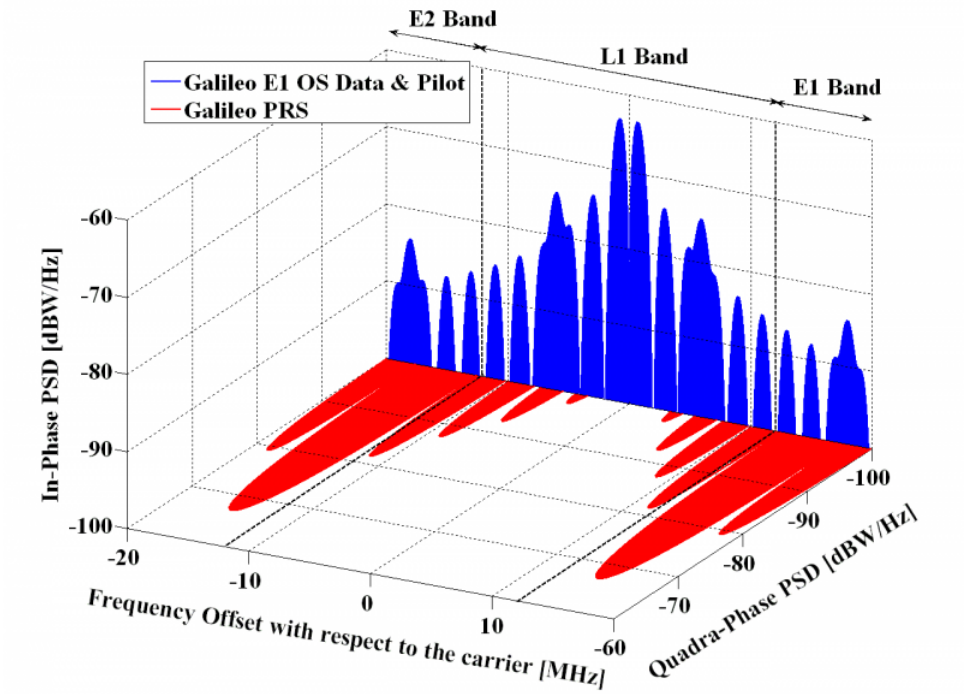


Figure 2.9: BOC (6,1,1/1) spectrum for the E1 signal (OS plus PRS). Source: Navipedia.

Figure 2.10 shows a generic view of the E1 OS signal generation while (2.3) and (2.4) describe the mathematical expressions for E1-C and E1-B signals respectively. Finally (2.5) shows the E1 BOC signal expression.

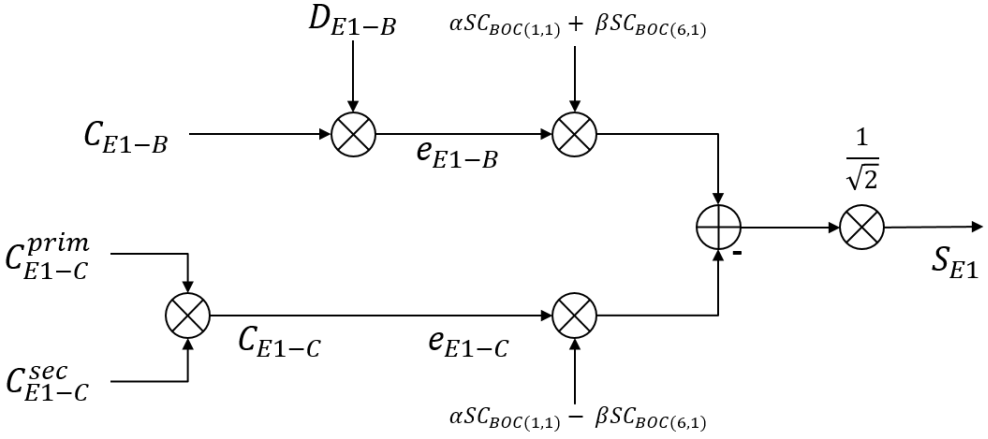


Figure 2.10: Block diagram for the E1 signal generation.

$$e_{E1-C}(t) = \sum_{m=-\infty}^{+\infty} C_{E1-Cs}\left(\mid m \mid_{25}\right) \oplus \sum_{l=-1}^{N} C_{E1-Cp}\left(l\right) \cdot p(t - mT_{c,E1-Cs} - lT_{c,E1-Cp}) \quad (2.3)$$

$$e_{E1-B}(t) = \sum_{l=-\infty}^{+\infty} C_{E1-B}\left(\mid l \mid_{N}\right) \oplus D_{E1-B}\left(l\right) \cdot p(t - lT_{c,E1-B}) \quad (2.4)$$

$$S_{E1}(t) = \frac{1}{\sqrt{2}}\left(e_{E1-B}(t)(\alpha sc_A(t) + \beta sc_B(t)) - e_{E1-C}(t)(\alpha sc_A(t) - \beta sc_B(t))\right) \quad (2.5)$$

Being $N = 4092$ the length of the Pseudo Random Noise (PRN) sequence (see Section 2.2.2.3) and the subcarriers and scaling factors described as:

$$\begin{aligned} sc_A(t) &= sign\left(\sin(2\pi f_{scA} t)\right) \\ sc_B(t) &= sign\left(\sin(2\pi f_{scB} t)\right) \\ \alpha &= \sqrt{10/11} \\ \beta &= \sqrt{1/11} \end{aligned} \quad (2.6)$$

2.2.2.2 Logic and power levels

For the whole Galileo system, the logic levels are the opposite to the signal level, which means that bit '1' will be represented by the level -1.0 and bit '0' by the level 1.0. This is also illustrated in Table 2.3.

Table 2.3: Logic to signal level assignment.

Logic level	Signal level
1	-1,0
0	+1,0

The total received minimum power (received by an ideally matched Right Hand Circular Polarization (RHCP), 0 dB gain antenna) is -157 dBW for the E1 signal.

2.2.2.3 Spreading codes

Regarding the E1 signal, E1-B and E1-C use different ranging codes.

- **E1-B** uses a 4 ms and 4092 chips long memory sequence, unique for each of the satellites. This means that the code rate is 1.023 MHz, the same as the main subcarrier frequency. Hence, for each chip period, one full subcarrier cycle is completed (high and low level).
- **E1-C** uses both a primary and a secondary code. Primary codes are unique for each satellite while the secondary code is common.

2.2.2.4 Navigation data

Although decodification of the navigation message is out of the scope of this project, some key concepts can be helpful:

1. The data rate is 250 sps.
2. This means that each 4 ms a symbol is transmitted (duration of one PRN sequence).
3. Therefore, every output of the correlation block can be seen as a single symbol.
4. For GPS, for example, one symbol lasts for 20 codes. See Appendix A.

Table 2.4 summarizes the characteristics of te Galileo signal.

Table 2.4: Summary of the Galileo E1-B signal characteristics.

Signal Component	E1 Data
Carrier Frequency	1575,42 MHz
Access Technique	CDMA
Modulation	CBOC (6,1,1/11)
Subcarrier Frequency	1,023 MHz and 6,138 MHz
Code Frequency	1,023 MHz
Code Length	4092 chips
Code Time length	4 ms
Data Rate	250 sps

2.3 RF front-end overview

Software signal processing of Global Navigation Satellite System (GNSS) signals requires input digital data. This is achieved using proper front ends. The following sections briefly explain the basic components of a GNSS front end.

2.3.1 Basic components

The main components of a front end are the antenna, filter, amplifier, mixer and analog-to-digital converter. The main ideas behind this section are taken from [9]. For further details, the reader is referred to [13] or [14].

Antenna The fundamental parameters are frequency, bandwidth, polarization and gain pattern. Two additional parameters can be considered: Voltage Standing Wave Ratio (WSWR) and impedance. Probably one of the most important in relation

to GNSS is the polarization parameter, which should be RHCP in order to be consistent with the transmitted signal from the satellite. Furthermore, if reflected off an object, the signal changes the polarization and the antenna will not be able to receive it. This is useful to mitigate multipath.

Filter The main reason to use a hardware filter before converting the signal to the digital domain is to eliminate any high power, out of band interference. Typically, the filter is the first element after the antenna. Important parameters of a filter are the bandwidth and the insertion loss (attenuation of the desired frequencies).

Amplifier The purpose of the amplification is to raise the signal level to a point where the Analog to Digital Converter (ADC) is able to perform the conversion. Therefore, the exact amplification needed is dependent on the chosen ADC.

Mixer Most ADCs can not operate at the E1 carrier frequency (1575.42 MHz). The local oscillator generates a tone and the mixer moves the signal to a frequency resulting of the difference between the carrier frequency and the local frequency (Intermediate Frequency (IF)). A copy of the signal is also generated at the sum of both frequencies, thus some filtering is needed after the mixer.

ADC The final component of the front end is the converter. The parameters to consider are the number of bits, sampling frequency, bandwidth and input range. For the particular case of GNSS signals, a couple of bits sampling could be used.

2.3.2 I/Q sampling

The mixer described before multiplies the RF signal by a sinusoidal signal (from the local oscillator). This way, the signal carrier is translated to IF: $f_{IF} = f_{RF} - f_{LO}$. Some mixers actually produce two outputs, multiplying the signal by a sin wave and a cosine wave respectively. Therefore, an in-phase component and a quadrature component are generated. Then the two signals are sampled independently generating a complex digital data stream. This process is called I/Q sampling [15].

The benefit of using I/Q sampling is that both the amplitude and phase of the signal can be recovered; in other words, the positive and negative parts of the spectrum.

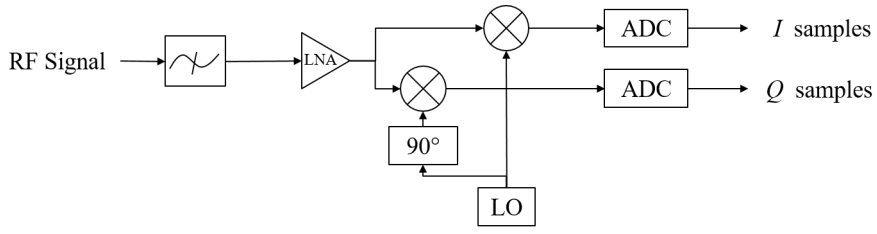


Figure 2.11: Basic front end block diagram for I/Q sampling.

2.4 Software Receiver Operations

The processing for GNSS systems is based on processing by channels, which applies to Galileo as well. Basically four operations are performed: acquisition, tracking, navigation data extraction and pseudorange calculation [10], although only acquisition and tracking are within the scope of this project.

1. **Acquisition**: the receiver has to know which satellites are visible. Either a warm start or a cold start can be performed. The acquisition process sweeps all the channels looking for satellites, and estimating the signal parameters.

2. **Tracking**: once satellites are acquired, fine carrier and code parameters have to continously be determined. This processes are called carrier and code tracking respectively.

3. **Navigation data extraction**

4. **Pseudorange calculation**

Figure 2.12 illustrates the receiver operations for one channel. The next sections describe the acquisition and tracking processes in detail.

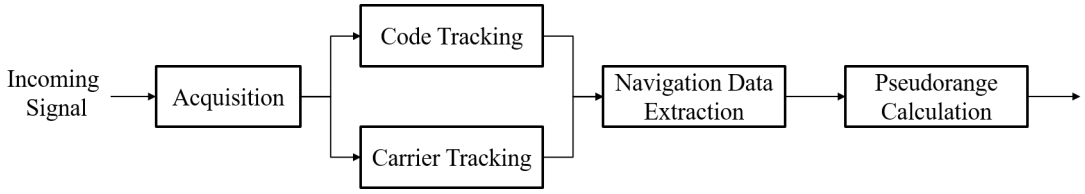


Figure 2.12: Receiver block diagram showing the main receiver operations [9].

2.4.1 Acquisition

As roughly mentioned before, the reason to perform acquisition is to let the receiver know which satellites are visible. If a satellite is visible, frequency and code phase have to be estimated.

Regarding frequency, signals are affected by the *Doppler effect*, which is translated in a small change in the frequency of the signal. For a stationary receiver on Earth, Doppler shift will not exceed 5 kHz [9]. The code phase refers to the data point where the PRN sequence begins. At least 4 ms are needed in order to ensure that the data includes one starting point.

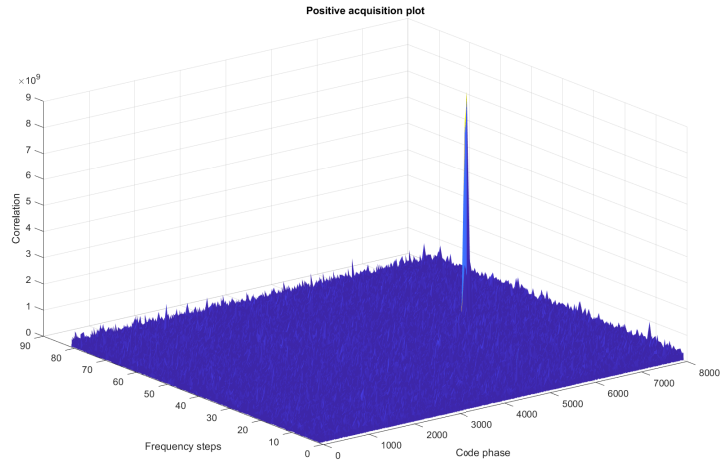


Figure 2.13: Positive acquisition plot (correlation sweep in frequency and code phase).

The idea behind acquisition is simple: the sampled signal is multiplied by a PRN sequence (locally stored memory code). This way cross correlation is being performed. Cross correlation between different codes has a very low impact. Sweeping for different frequencies and code phase points, a peak on the cross correlation should appear when the incoming signal and the generated code are using the same frequency and the correct code phase.

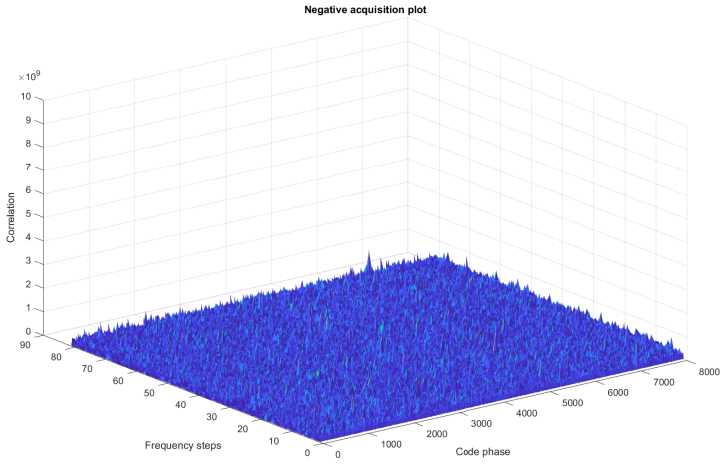


Figure 2.14: Negative acquisition plot (correlation sweep in frequency and code phase).

Using frequency steps of 500 Hz (21 points), 4092 different code phases and 30 satellites, all the frequencies for every code phase and every satellite will have been checked. If there is a peak and it is over a certain threshold, the satellite is acquired. Figure 2.13

shows a typical acquisition plot for an acquired satellite and Figure 2.14 the same plot for a not present satellite.

However, parallel code phase search can be performed to minimize the number of correlations, performing a Fourier transform of the signal and the local PRN sequence, and multiplying in the frequency domain. Afterwards, an inverse Fourier transform is performed and the output is evaluated. The block diagram is shown in Figure 2.15.

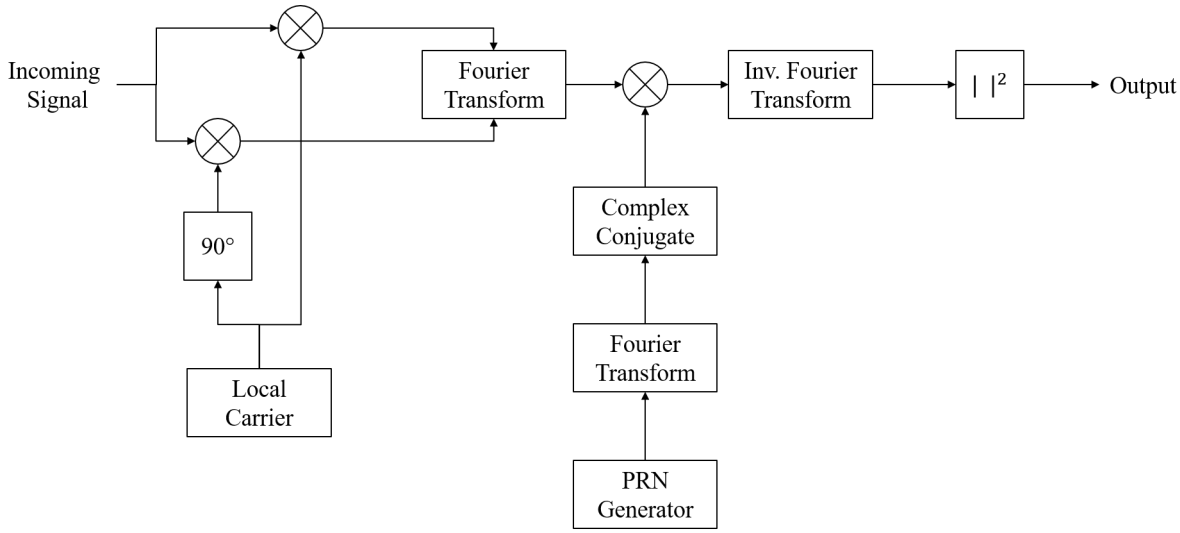


Figure 2.15: Parallel acquisition block diagram.

More details about parallel acquisition are explained in Section 3.3.

2.4.2 Tracking

The main reason to perform tracking is to get a better and continuos estimation for the parameters obtained during the acquisition (code phase and frequency), as well as to keep track of these during the whole reception process.

Therefore, two different algorithms take part on the tracking:

1. **Code Tracking**: Often implemented as a Delay Lock Loop (DLL), where an early and a late replica of the code are correlated with the incoming signal. The two outputs are compared by the discriminator and the system adjusts to perform sampling in the right time. The block diagram for the code tracking is shown in Figure 2.16.

2. **Carrier Tracking**: Can be done either by tracking the phase of the signal or the frequency. Often implemented using a Phase Lock Loop (PLL), usually a Costas

Loop where two samples (in-phase and quadrature) are compared and a Numeric Controlled Oscillator (NCO) is adjusted to keep the energy on the in-phase arm. The block diagram for the carrier tracking is shown in Figure 2.17. The Costas loop is used because it is non-sensitive to 180 deg. phase changes, thus non-sensitive to phase transitions due to navigation bits [9].

These two algorithms can be combined in a single one to save computational resources. Further information about tracking is given in Section 3.4.

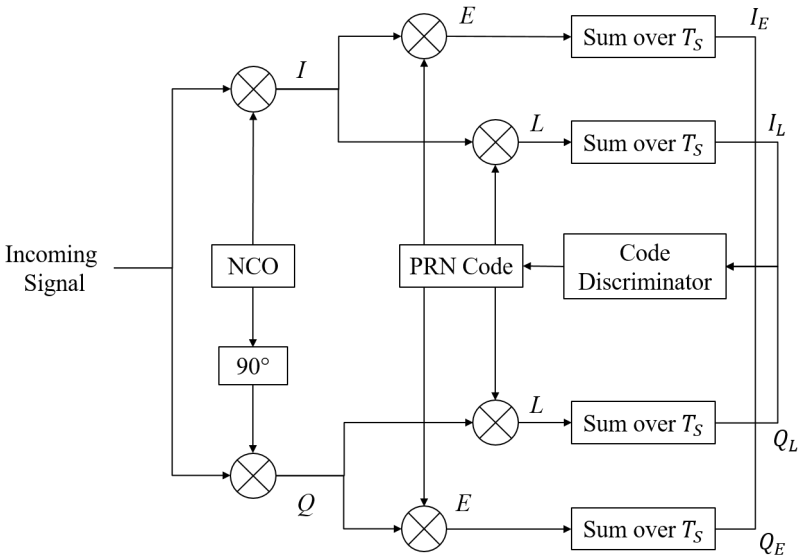


Figure 2.16: DLL block diagram used for code tracking.

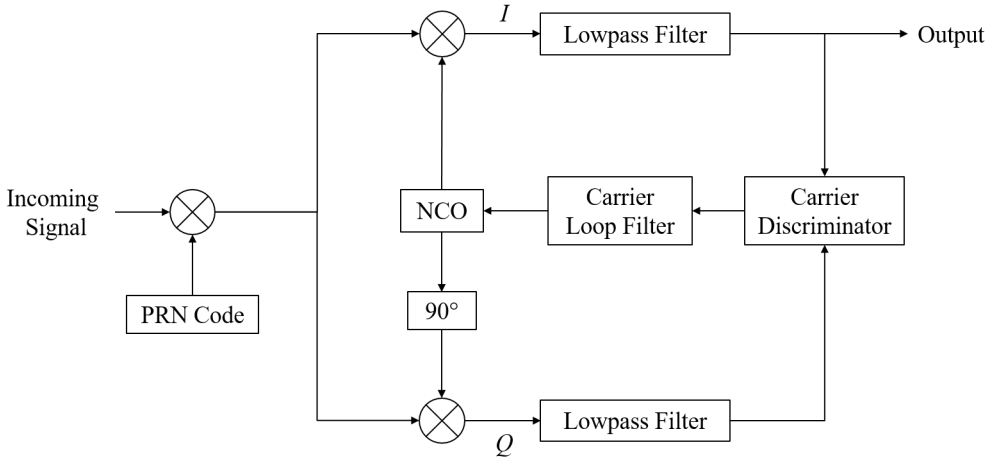


Figure 2.17: Costas loop block diagram used for carrier tracking.

CHAPTER 3

Implementation

3.1 Software toolbox

The Galileo toolbox presented in this project is based on an existing GPS toolbox [6], where some modifications have been performed according to the changes of the Galileo signal with respect to the GPS signal (See Appendix A for further details about the GPS signal).

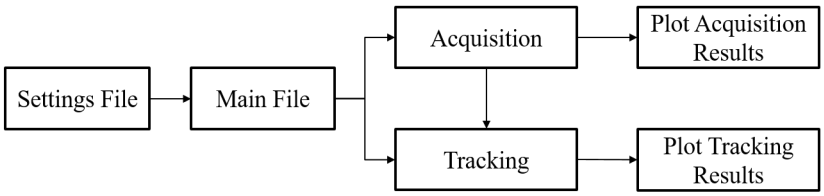


Figure 3.1: General block diagram of the software toolbox operations.

The only input that the toolbox requires in order to work is a GNSS signal consisting of 8 bit IF samples. It is also necessary to know the main parameters of the signal (sampling frequency, samples format, etc.). More details about data collection and data types are given in Section 4.1 and Section 4.2 respectively.

The next sections describe the main blocks of the toolbox: main file, acquisition and tracking algorithms. The toolbox is written using Matlab, although some specific functions are implemented in C.

3.2 Main file and settings

Basically the whole toolbox is governed by a main file which receives the settings from a settings file and launches the acquisition and tracking algorithms. Finally, some results are plotted. Figure 3.1 shows a block diagram of the toolbox.

The settings file contains some important data regarding the signal and also all the acquisition and tracking parameters:

Signal parameters

- Directory name
- File name
- Samples format (I or I/Q)
- Sampling frequency
- Intermediate frequency
- Front end bandwidth

Common parameters

- Code frequency
- Code Length
- Subcarrier frequency

Acquisition parameters

- Search frequency band
- Satellites list
- Threshold

Tracking parameters

- ms to process
- PLL bandwidth
- DLL bandwidth
- PLL order
- DLL order

Once this settings file is configured, the main file loads all the parameters and launches the acquisition. When the acquisition has finished, the results are saved and shown. Then tracking is launched: first a tracking struct is initialized and then every channel is tracked in parallel. Finally the tracking results are plotted. The basic operations are described in Algorithm 1.

Algorithm 1 Main file

1: Given: settings file containing all the necessary parameters.
2: Load *settings*
3: $aqc_{results} \leftarrow acquisition(settings)$ ▷ Launch acquisition
4: Save $aqc_{results}$
5: Initialize struct $track_{results}$ ▷ Save space for every acquired satellite's results and assign BOC sequences to track
6: **for** *acquired satellites* **do** ▷ Launched in parallel (independent channels)
7: $track_{results} \leftarrow trackChannel(acq_{results}, settings)$ ▷ Launch tracking
8: **end for**
9: $plotTracking(track_{results})$

3.3 Galileo Acquisition

As seen in Figure 2.15, acquisition can be done in parallel in order to make the algorithm more efficient. This way, the computational cost is reduced by a factor of 4092, since the sweep in code phase is no longer necessary. Therefore, all the frequencies must be tested for every satellite (every code, since different ones are assigned to satellites).

Apart from the incoming GNSS signal, which is stored in memory, two other input signals are necessary: the local carrier and the local PRN sequences (CBOC signals).

3.3.1 Carrier generation

Actually two signals have to be generated, with a phase difference of 90 deg. (a cosine and a sine wave). The frequency of those has to be the IF (plus or minus the Doppler shift being tested) and the digital signal must be sampled according to the ADC sampling frequency. It should have a length of 4 ms (one PRN sequence). A block diagram showing the inputs and outputs is shown in Figure 3.2 and the main steps are illustrated in Algorithm 2.

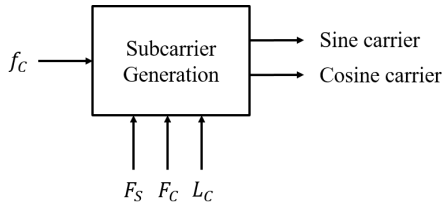


Figure 3.2: Carrier generation block inputs and outputs.

Algorithm 2 Carrier generation

1: Given: desired carrier frequency f_C, sampling frequency F_S, code frequency $F_C = 1.023 \cdot 10^6$, code length $L_C = 4092$.
2: $N_S \leftarrow \frac{F_S}{F_C} L_C$ ▷ Calculate number of samples per code
3: $T_S \leftarrow 1/F_S$ ▷ Calculate sampling period
4: $Sin_C \leftarrow \sin(2\pi f_C T_S[0 : N_S - 1])$ ▷ Generate sine carrier
5: $Cos_C \leftarrow \cos(2\pi f_C T_S[0 : N_S - 1])$ ▷ Generate cosine carrier
6: **return** Cos_C, Sin_C

3.3.2 BOC generation

PRN sequences for the Galileo system are 4092 chips long, with a rate of 1.023 Mcps, which means that the codes last 4 ms. However, as Figure 2.6 shows, this code has to be multiplied by a signal which is a combination between two sqare subracciers.

In order to implement a narrowband receiver as simple as possible, only the first subcarrier will be considered (1.023 MHz) while the other will simply be removed (6.138 MHz). Therefore, each chip is multiplied by one subcarrier period, meaning the CBOC signal will have two values for each input chip. This CBOC signal has 8184 values and lasts for 4 ms, thus the rate is 2.046 Mcps.

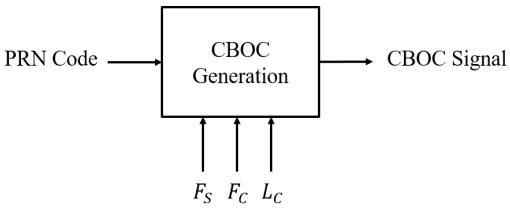


Figure 3.3: CBOC generation block inputs and outputs.

The composite signal generated can be seen as a new PRN code with twice the length and the frequency of the original one. Of course this new code has to be sampled according to the sampling frequency. A block diagram showing the inputs and outputs is shown in Figure 3.3 and the CBOC generation algorithm is shown in Algorithm 3.

Algorithm 3 BOC generation

1: Given: PRN code for the desired satellite C_B, sampling frequency F_S, code frequency $F_C = 1.023 \cdot 10^6$, code length $L_C = 4092$.
2: $C_B(C_B == 1) \leftarrow -1$ ▷ Signal value assignnment for a '1' bit
3: $C_B(C_B == 0) \leftarrow +1$ ▷ Signal value assignnment for a '0' bit
4: $BOC \leftarrow C_B(duplicated\ each\ value)$
5: $BOC(even\ samples) \leftarrow -BOC(even\ samples)$ ▷ Flip the second sample for each chip – simulating the subcarrier effect
6: $N_S \leftarrow \frac{F_S}{F_C} L_C$ ▷ Calculate number of samples per code
7: $i_{resample} \leftarrow 2\frac{F_C}{F_S}[1 : N_S]$ ▷ Build a resample index from 1 to 8184 with the number of samples per code
8: $CBOC \leftarrow BOC(i_{resample})$ ▷ Generate sampled BOC signal
9: **return** $CBOC$

3.3.3 Acquisition algorithm

Once the local carrier and the BOC signals can be generated, all the inputs are available for the acquisition block (see Figure 3.4). It basically consists of two loops: one to sweep over the 30 satellites (30 different BOC sequences) and the other to sweep over different Doppler frequency shifts.

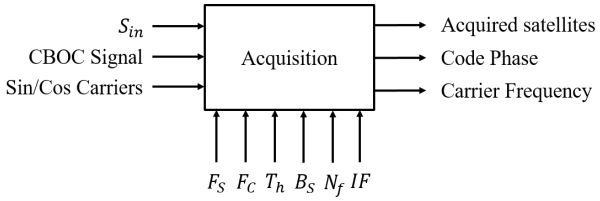


Figure 3.4: Acquisition block inputs and outputs.

As illustrated on Figure 2.15, the signal is multiplied by the carriers and then transformed to the frequency domain, where it is multiplied by the CBOC sequence. After performing the inverse Fourier transform, the result of the correlation is evaluated and if the peak is above a certain threshold, the satellite is acquired and two parameters are stored: the code phase and the carrier frequency. The algorithm is explained in Algorithm 4.

Algorithm 4 Acquisition

Given: number of Doppler frequencies N_f, intermediate frequency IF, search band BW_S, incoming complex signal (I/Q) S_{in}, sampling frequency F_S, code frequency $F_C = 1.023 \cdot 10^6$, acquisition threshold Th
for PRN = 1:30 **do** ▷ Every satellite
  $CBOC \leftarrow$ corresponding BOC signal from Algorithm 3
  $CBOC_{freq} \leftarrow conj(fft(CBOC))$
  for $i_F = 1 : N_f$ **do** ▷ Every Doppler frequency step
    $freq(i_F) \leftarrow IF - BW_S/2 + 50(i_F - 1)$ ▷ Calculate frequency to examine
    Generate Cos_C and Sin_C from Algorithm 2
    $I \leftarrow Cos_C \cdot real(S_{in}) + Sin_C \cdot imag(S_{in})$ ▷ Calculate in-phase signal
    $Q \leftarrow -SinC \cdot real(S_{in}) + Cos_C \cdot imag(S_{in})$ ▷ Calculate quadrature signal
    $IQ_{freq} \leftarrow fft(I + jQ)$ ▷ Transmorm signal to the frequency domain
    $Conv \leftarrow IQ_{freq} \cdot CBOC_{freq}$ ▷ Perform multiplication in frequency domain
    $Acq(i_F) \leftarrow abs(ifft(Conv))^2$ ▷ Transform to the time domain
  end for
  $[peak,\ carrFreq,\ codePhase] \leftarrow \max(Acq)$
  $N_C \leftarrow \dfrac{F_S}{F_C}$ ▷ Find samples per code chip
  $exc_1 \leftarrow codePhase - N_C$ ▷ Exclude one chip before the peak
  $exc_2 \leftarrow codePhase + N_C$ ▷ Exclude one chip after the peak
  $peak_2 \leftarrow \max(Acq, [1 : exc_1, exc_2 : end])$ ▷ Find second peak
  if $\dfrac{peak}{peak_2} > Th$ **then** ▷ If the ratio is above the threshold, save the values
    Mark satellite as acquired
    $Acq_{results}(PRN) \leftarrow codePhase,\ carrFreq,\ peak$
  else
    Mark satellite as not acquired
  end if
end for
return $Acq_{results}$

3.3.4 Fine resolution frequency search

In theory, the output from the acquisition block should be used as input for the tracking algorithms. However, it is interesting to perform a *fine resolution frequency search*.

The idea is to multiply the GNSS signal with the correspondent CBOC sequence but now taking into account the proper code phase obtained before. Furthermore, instead of the duration of one sequence (4 ms), the operation will be performed over a multiple of the duration of one code (for example, 16 ms).

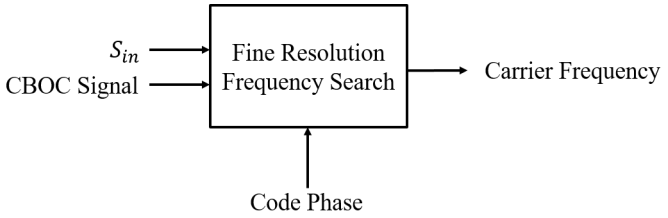


Figure 3.5: Fine resolution frequency search block inputs and outputs.

The output is transformed to the frequency domain and the carrier frequency is obtained by finding the maximum value. A block diagram showing the inputs and outputs to this fine resolution frequency search block is shown in Figure 3.5 and Algorithm 5 illustrates the process.

Algorithm 5 Fine resolution frequency search

1: Given: code phase $codePhase$, BOC sequence BOC, incoming complex signal (I/Q) S_{in}.
2: $BOC_{long} \leftarrow [BOC\ BOC\ BOC\ BOC]$ ▷ Concatenate 4 times the BOC sequence
3: $X_{carrier} \leftarrow BOC_{long} \cdot S_{in}(codePhase : end)$ ▷ Remove the BOC sequence from the signal (using the proper code phase)
4: $X^{freq}_{carrier} \leftarrow fft(X_{carrier})$
5: $[\sim, carrierFreq] \leftarrow max(X^{freq}_{carrier})$ ▷ Select the maximum value in the frequency domain
6: **return** $carrierFreq$

Therefore, the value for the carrier frequency used for the tracking is the one obtained from the fine resolution algorithm while the code phase value used is the one obtained from the acquisition block.

3.4 Galileo Tracking

Figure 2.16 and Figure 2.17 show the code and carrier tracking loops respectively. However, those can be implemented simultaneously (saving time and computational resources). The final implemented block diagram is illustrated in Figure 3.6.

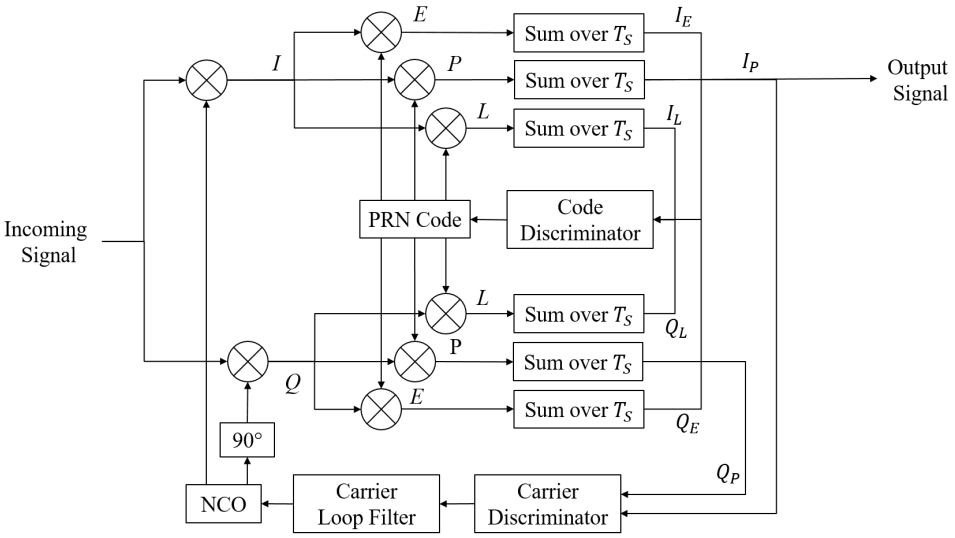


Figure 3.6: Tracking block diagram joining carrier and code tracking loops.

The algorithm tracks one Galileo channel and the main blocks are the discriminators (code and carrier), the PRN and carrier generators and of course the correlators (6 different outputs: in-phase and quadrature, early, late and prompt). The CBOC sequence is generated in a similar way as done in the acquisition and the same applies to the carrier generation. The tracking algorithm is shown in Algorithm 6.

Algorithm 6 Tracking

Given: empty results struct $trackData$, ms to process ms, correlator spacing SPC, BOC sequence BOC, BOC length $L_{BOC} = 2 \cdot 4092$, initial code phase Ph_{code}, initial carrier phase $Ph_{carrier}$, sampling frequency F_S, code frequency F_C, carrier frequency f_C, incoming complex signal (I/Q) S_{in}.
$j \leftarrow 1$; ▷ Initialize index
while $j < ms + 1$ **do**
    $Blk_{size} \leftarrow \frac{L_{BOC} - Ph_{code}}{F_C} F_S$ ▷ BOC sequence lasting samples size
    $[IE, IP, IL, QP, QE, QL, Ph_{code}, Ph_{carrier}] \leftarrow$ ▷ Perform correlation
    $\leftarrow correlation(S_{in}, BOC, Blk_{size}, F_S, F_C, Ph_{code}, Ph_{carrier}, f_C, SPC)$
    $Ph_{code} \leftarrow Ph_{code} - L_{BOC}$ ▷ Reset code phase
    $Ph_{carrier} \leftarrow rem(Ph_{carrier}, 2\pi)$ ▷ Reset carrier phase
    $C_{offset} \leftarrow C_{offset} + Blk_{size}$ ▷ Update offset
    $f_C \leftarrow PLL(IP, QP)$ ▷ Calculate carrier frequency (PLL discriminator)
    $F_C \leftarrow DLL(IE, IL, QE, QL)$ ▷ Calculate code frequency (DLL discriminator)
    $trackData \leftarrow IE, IP, IL, QP, QE, QL, Ph_{code}, Ph_{carrier}, f_C, F_C$ ▷ Save data
    $j \leftarrow j + 4$ ▷ Index is incremented in 4 ms (duration of the BOC sequence)
end while
return $trackData$

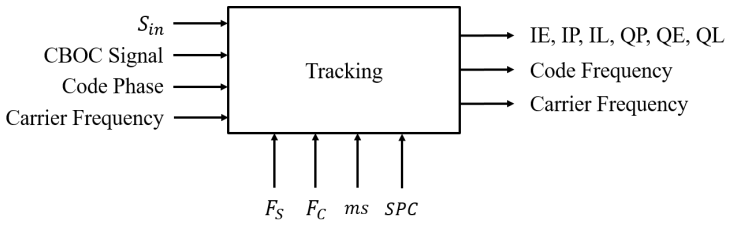


Figure 3.7: Tracking block inputs and outputs.

Figure 3.7 shows the inputs and outputs of the tracking block. Section 3.4.1 and Section 3.4.2 explain the implementation of the remaining blocks inside the tracking loop: the correlation fuction and the DLL and PLL discriminators.

3.4.1 Correlation function

The correlation function is probably the most resource consuming block of the whole system. Therefore, it is implemented in C and compiled as a *mex* file to run in Matlab. It calculates the in-phase and cuadrature, early, late and prompt outputs for every iteration of the tracking loop described in Algorithm 6. It also outputs the carrier phase and the code phase to be used in the next iteration.

Algorithm 7 describes the correlation algorithm and Figure 3.8 shows the inputs and outputs of the correlation block.

Algorithm 7 Correlation

Given: correlator spacing SPC, BOC sequence BOC, block size Blk_{size}, initial code phase Ph_{code}, initial carrier phase $Ph_{carrier}$, sampling frequency F_S, code frequency F_C, carrier frequency f_C, incoming complex signal (I/Q) S_{in}.
$IE, IP, IL, QP, QE, QL \leftarrow 0$; ▷ Initialize outputs
for $i = 0 : 2 : 2Blk_{size}$ **do** ▷ Loop for every two samples of the block (I/Q)
    $I_{data} \leftarrow -S_{in}(i) \cdot \cos(Ph_{carrier}) + S_{in}(i+1) \cdot \sin(Ph_{carrier})$
    $Q_{data} \leftarrow S_{in}(i) \cdot \sin(Ph_{carrier}) + S_{in}(i+1) \cdot \cos(Ph_{carrier})$
    $IE \leftarrow IE + I_{data} \cdot BOC(Ph_{code} - SPC)$
    $IP \leftarrow IP + I_{data} \cdot BOC(Ph_{code})$
    $IL \leftarrow IL + I_{data} \cdot BOC(Ph_{code} + SPC)$
    $QE \leftarrow QE + Q_{data} \cdot BOC(Ph_{code} - SPC)$
    $QP \leftarrow QP + Q_{data} \cdot BOC(Ph_{code})$
    $QL \leftarrow QL + Q_{data} \cdot BOC(Ph_{code} + SPC)$
    $Ph_{carrier} \leftarrow Ph_{carrier} + \frac{2\pi f_C}{F_S}$ ▷ Update carrier phase for next iteration
    $Ph_{code} \leftarrow Ph_{code} + \frac{F_C}{F_S}$ ▷ Update code phase for next iteration
end for
return $IE, IP, IL, QP, QE, QL, Ph_{carrier}, Ph_{code}$

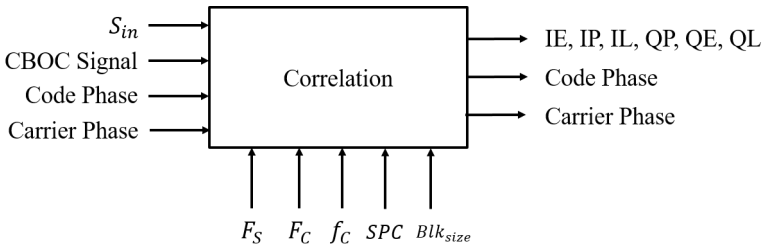


Figure 3.8: Correlation block inputs and outputs.

3.4.2 DLL and PLL discriminators

In every iteration of the tracking loop, after the in-phase and cuadrature, early, late and prompt outputs have been calculated by the correlation block, the DLL and PLL discriminators output the carrier frequency and code frequency to use in the next iteration respectively. These values are based on the discriminator functions.

PLL discriminator

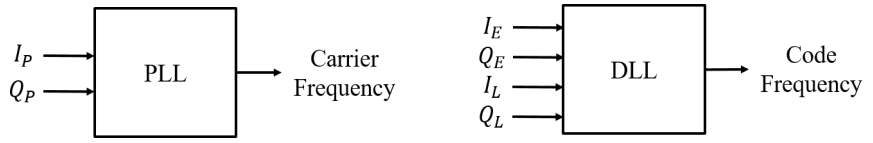
The function is the inverse tangent of the ratio between the prompt Q and I values (3.1). The discriminator outputs the phase error, which is filtered by the carrier loop filter to obtain the carrier frequency. Figure 3.9(a) shows the block inputs and outputs.

DLL discriminator

The function is a normalized early minus late power noncoherent discriminator (3.2), which also outputs the phase error, filtered to obtain the code frequency. Figure 3.9(b) shows the block inputs and outputs.

$$D_{PLL} = \tan^{-1}\left(\frac{Q_P}{I_P}\right) \tag{3.1}$$

$$D_{DLL} = \frac{\sqrt{(I_E^2 + Q_E^2)} - \sqrt{(I_L^2 + Q_L^2)}}{\sqrt{(I_E^2 + Q_E^2)} + \sqrt{(I_L^2 + Q_L^2)}} \tag{3.2}$$



(a) PLL block inputs and outputs (b) DLL block inputs and outputs

Figure 3.9: DLL and PLL blocks inputs and outputs.

Figure 3.10 shows the general picture of the system processes. The main file loads the settings and launches the acquisition. When the results are available, the acquired channels must be tracked in parallel. The main script launches the tracking loop and receives the results, that are finally plotted after every satellite is tracked.

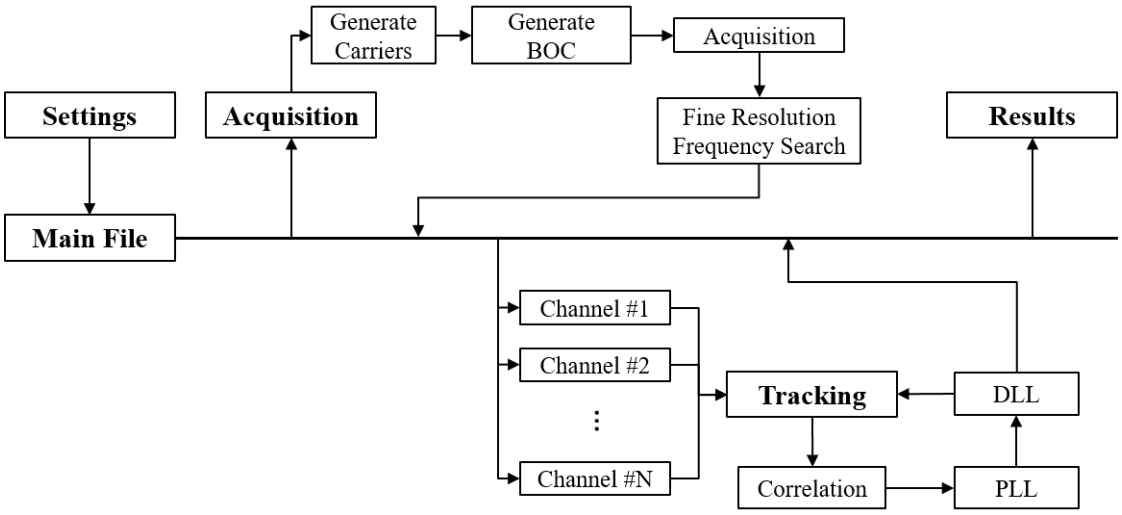


Figure 3.10: Full system timeline showing the different blocks and sub-blocks.

CHAPTER 4

Results and performance

After the signal has been studied and a software tracking toolbox has been developed, the necessity of testing appears. Some IF signal records can easily be found in the internet, but there are some reasons to think about performing new data collection.

- First, the Galileo constellation is not complete yet, and new satellites are being launched.
- Second, the signal characteristics have changed since the first testing satellites were launched.
- Finally, it is very interesting to be able to test the toolbox using different signals with different parameters: sampling frequency, filter bandwidth, data types, local oscillator reference, etc.

4.1 Data collection set-up

Many options are available to collect IF data at the GNSS frequency, most of them consisting of a tunable SDR receiver. For this purpose, the chosen device is a USRP (USRP B200 mini), designed and sold by Ettus Research (National Instruments). The characteristics of this receiver are explained in Appendix B.

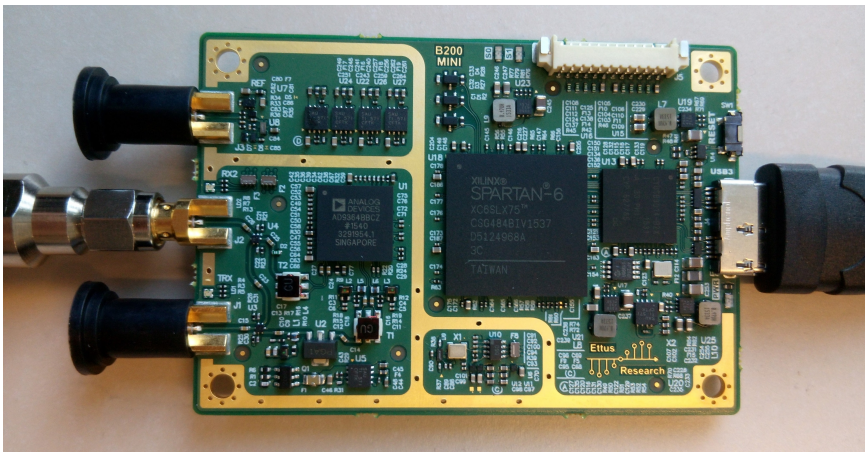


Figure 4.1: USRP board used for data collection.

Figure 4.1 shows the board used. Basically the USRP module has three ports: transmission, reception and reference, and a USB interface to connect to a PC. When a GNSS antenna is connected to the reception port, samples can be recorded. The recording is performed using GNURadio (basic configuration explained in Appendix C), which is a free software used to tune the receiver: carrier frequency, sampling frequency, filter bandwidth, etc. The layout used for Galileo data collection is shown in Figure C.1.

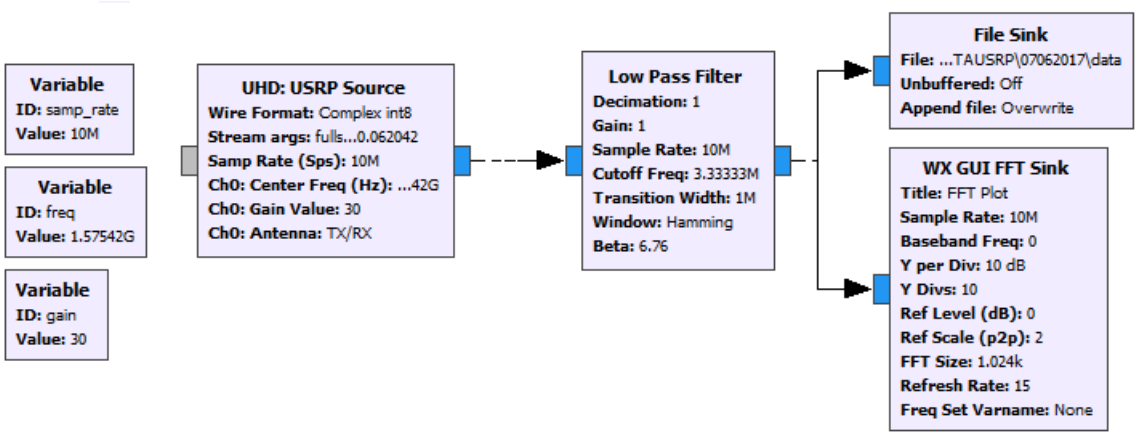


Figure 4.2: GNU Radio flowchart for IF data collection.

Finally, since the toolbox works with 8 bit samples, some data conversion has to be made. The script for data conversion is shown in Appendix D.

4.2 Signal sets tested

Even though an environment for data collection has been set up, it is also interesting to test the toolbox using simulated data, just because the signal is clean and this makes debug easier. Thus, both simulated and real signals have been tested.

Simulated data Two simulated signal sets have been used, obtained from the European Space Agency (ESA) and Institute Of Navigation (ION) respectively. This is the first step to prove that the system is working, since the satellites that are present in the signal are known in advance.

Real data Once the toolbox is working with simulated signals, it can be tested with real GNSS samples. Since the collection set-up described in Section 4.1 has not been tested, it is interesting to use an externally collected signal before, known to contain GNSS signals. For this purpose, an external signal set has been used, obtained from the ESA and containing GIOVE-A signals (previously known as Galileo System Testbed (GSTB)). Apart from this set, the toolbox has also been tested with data collected using a sampler built at DTU Space (for GPS).

Collected data Once the toolbox is working using simulated and real data, the collection set-up can also be analyzed in order to perform our own signal recording. The antenna is positioned at the roof top of the DTU Space building. Different sampling frequencies and filter bandwidths have been tried.

The following sections describe the performance of the acquisition and tracking using the previously mentioned signal sets.

4.3 Performance

The acquisition algorithm has proved to be very robust. This was expected since it is only a coarse estimation of the signal parameters. For every data set tested, a peak metric plot has been obtained, showing the value of the correlation peaks (shown on Section 2.4.1, Figure 2.13). Besides this, the power spectral density of every signal is also plotted.

The tracking function is able to provide more details about the process. As a consequence of this, not only the in-phase and quadrature outputs are shown but other parameters too:

1. **Discriminators' output**: Every correlation period (4 ms for Galileo), values for the code frequency and carrier frequency errors out of the DLL and PLL are stored. These values are plotted and should tend to zero for a tracked satellite.

2. **Code and carrier frequencies**: for a tracked satellite they should remain stable after the first fluctuations.

    - *Code frequency*: as mentioned in Section 3.4.2, the error from the DLL is filtered to obtain the corresponding code frequency. This value should be around the the Galileo code frequency (1.023 MHz). However, it is interesting to notice that the PRN code together with the main subcarrier (also 1.023 MHz, two values per code chip) is understood as a new code sequence with 2.046 MHz frequency. Therefore, this will be the reference value for the code tracking algorithm.
    - *Carrier frequency*: in a similar way as the DLL outputs the frequency, the PLL also filters the carrier frequency error to obtain the carrier frequency. The Doppler shift is assumed to be less than 10 kHz, thus the value of the carrier frequency must be within 0 and 10 kHz.

3. **Prompt correlators' output**: The objective of the Costas loop is to keep all the energy in the in-phase component. Therefore, it is expected that the output of the prompt quadra-phase correlator will tend to zero while the output of the prompt in-phase correlator will show the tracked signal: BPSK symbols.

4.3.1 Simulated data

Due to the fact that the signal has not been collected from a satellite but simulated locally, eight satellites are acquired (see Figure 4.3). It can easily be identified as simulated data: for example satellite with PRN 23 is acquired but it has not been launched yet.

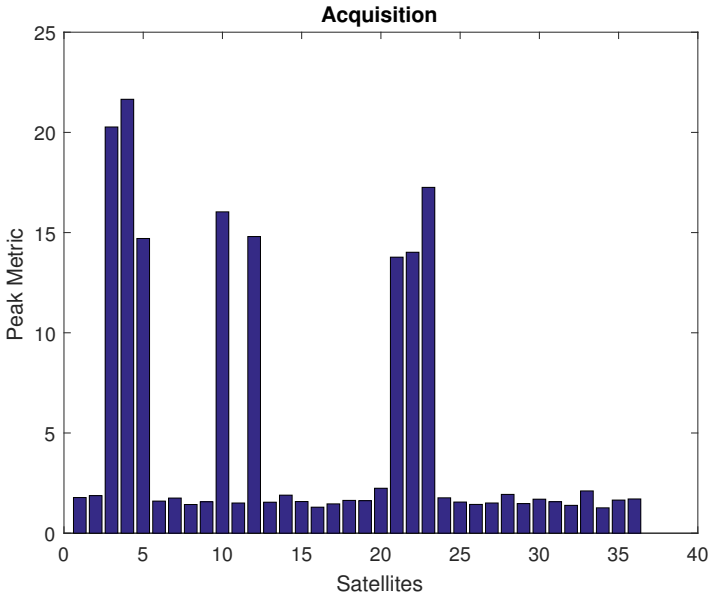
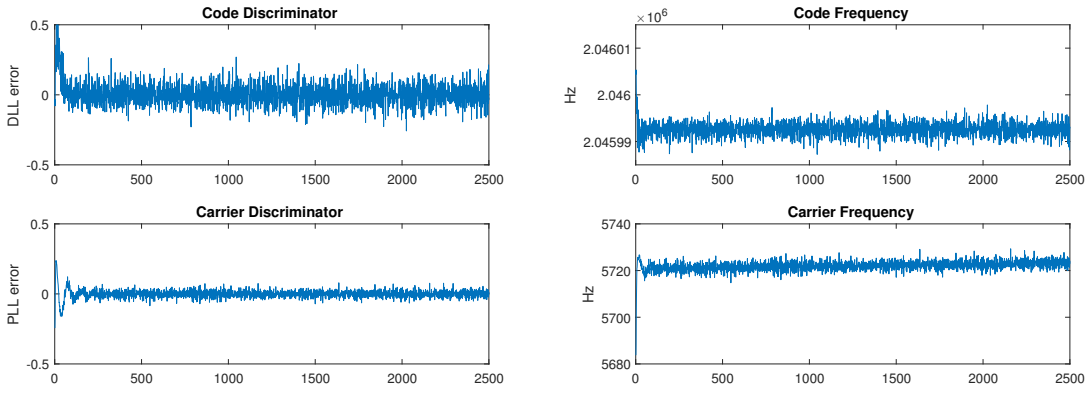


Figure 4.3: Simulated signal acquisition plot.

The tracking data plotted in this section is obtained for PRN 23. Figure 4.4(a) and Figure 4.4(b) show the discriminators' output and the code and carrier frequencies respectively.



(a) Simulated signal discriminators' output (b) Simulated signal code and carrier frequencies

Figure 4.4: Simulated signal discriminators' output (a), code & carrier frequencies (b).

The DLL and PLL errors fluctuate around zero and the frequencies seem to be stable along the whole tracking time, which means that the tracking is correct. The correlators' output plot shows the energy in the in-phase arm in form of symbols (Figure 4.5).

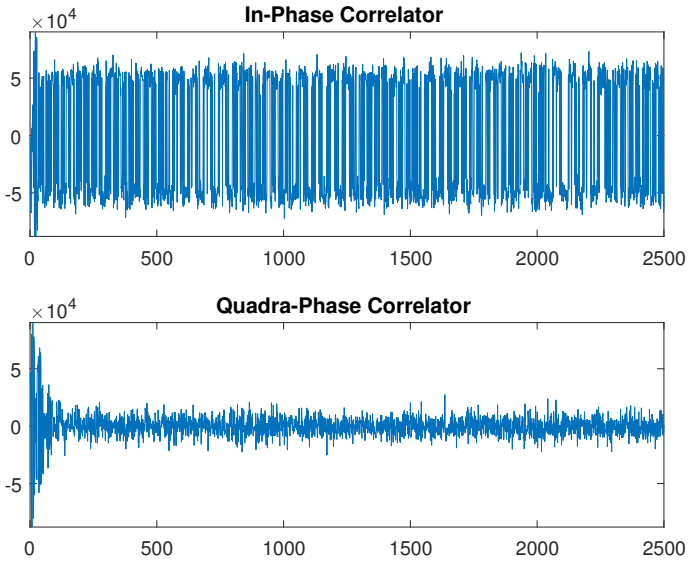


Figure 4.5: Simulated signal prompt correlators' output.

Figure 4.6 shows a zoomed part of the correlators' output where it is clear that the in-phase data is representing bits while the quadra-phase data is just noise around zero.

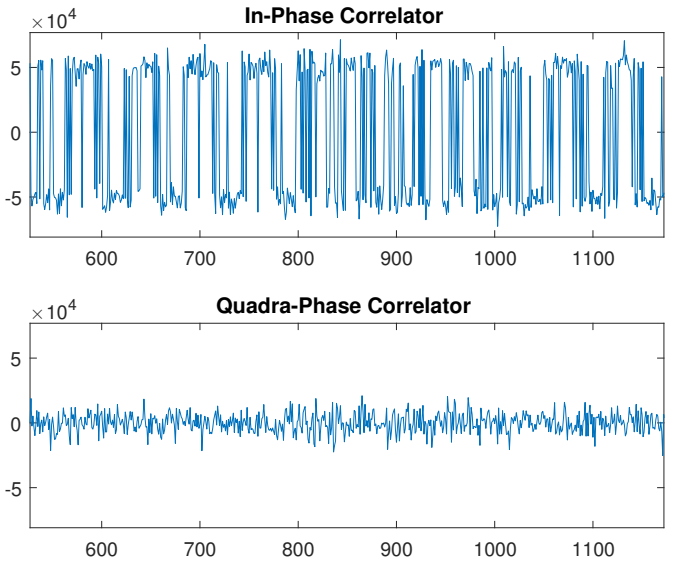


Figure 4.6: Simulated signal prompt correlators' output zoomed.

4.3.2 Real data

In the previous section, the toolbox has been proved to be working for Galileo signals. However, it is necessary to test it with real signals. For this purpose, the data set collected with the DTU Space sampler for GPS has been tested. Two satellites are acquired: PRN 2 and PRN 8, both in orbit (Figure 4.7).

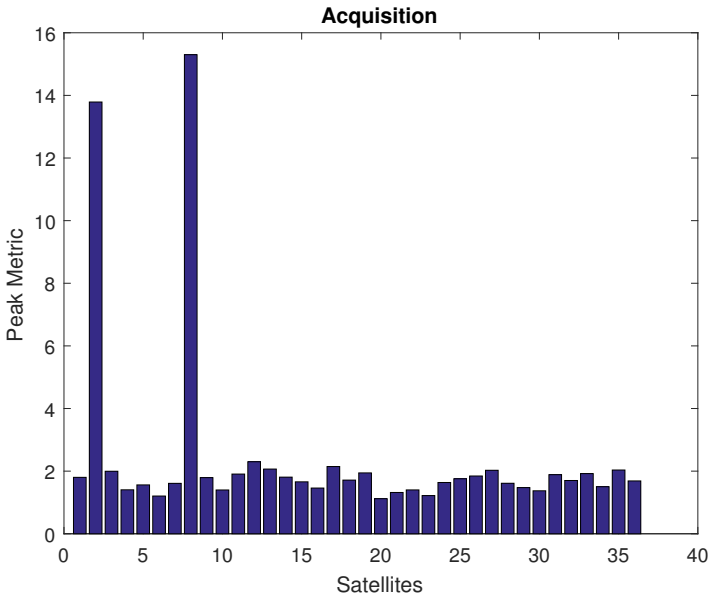
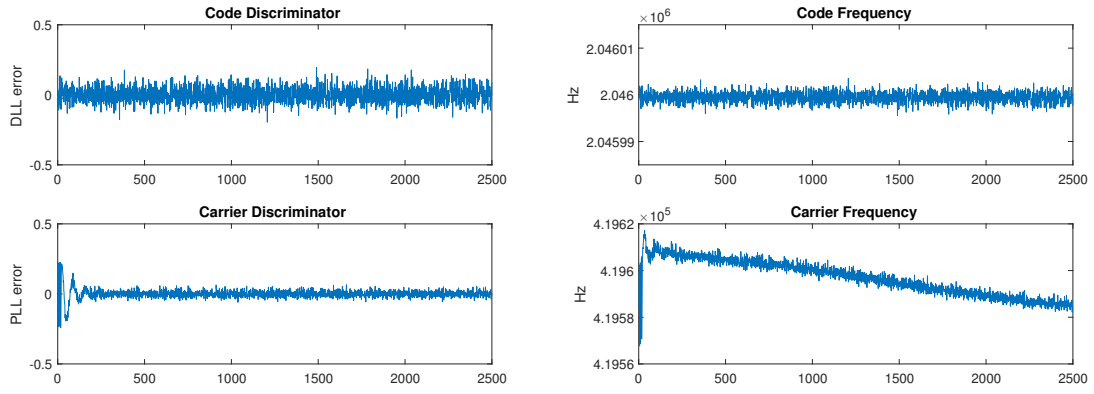


Figure 4.7: Real signal acquisition plot.

Satellite with PRN 8 is analyzed. The discriminators' output and the frequencies seem to be according to what it is expected (Figure 4.8(a), Figure 4.8(b)).



(a) Real signal discriminators' output

(b) Real signal code and carrier frequencies

Figure 4.8: Real signal discriminators' output (a), code & carrier frequencies (b).

Figure 4.9 shows the correlators' output for this satellite, which confirms the correct tracking.

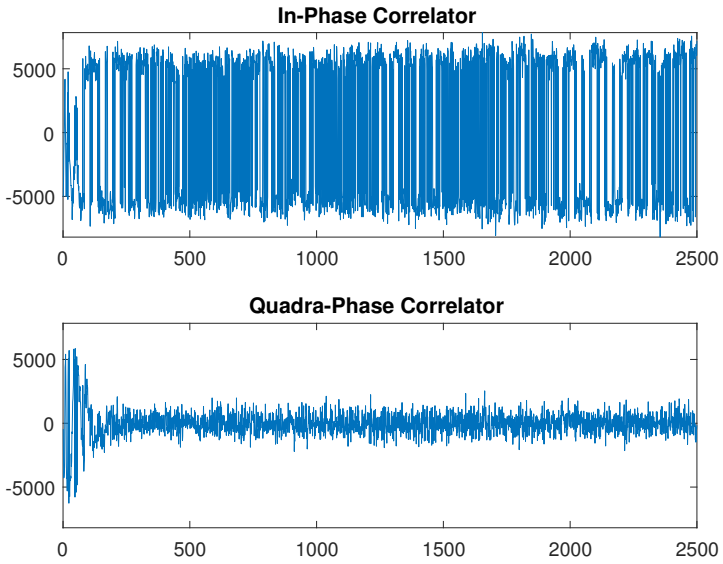


Figure 4.9: Real signal prompt correlators' output.

Some differences are observed with respect to the simulated data set:

1. The code frequency seems to be more accurate (close to the reference value 2.046 MHz). The reason for this to happen can be the precision of the reference clock. The satellite and receiver clocks can be more precise and accurate than the one used to simulate the signal.

2. The carrier frequency seems to be changing over time. This has a reasonable and simple explanation: the satellite is moving at a certain speed and due to the *Doppler effect*, the frequency changes. In this case, the frequency decreases over time, which means that the satellite is moving away from the receiver.

4.3.3 Collected data (USRP)

The two previous sections show the system working for both simulated and real Galileo signals. Therefore, it should be able to work with data collected with the USRP module.

For this test, the signal is coming from an antenna situated on the roof top of the building, meaning it should be free from reflections from the ground or buildings. Besides this, the antenna is an active antenna, where some amplification is performed before the signal is handled by the receiver. For this reason, the gain has been adjusted to 30 dB.

Figure 4.10 shows the acquired satellites. In this case, satellites with PRN 3, PRN 5 and PRN 24 are acquired. Notice the high peak metric of the closest satellite, which is a consequence of the good quality of the collected data.

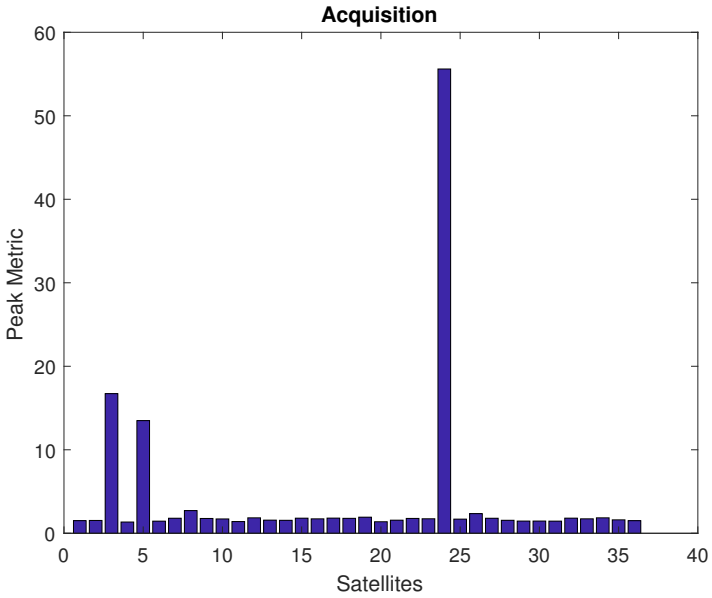


Figure 4.10: USRP signal acquisition plot.

In order to verify that the signal is correct, Figure 4.11 shows a live view of the Galileo satellites at the moment of the data collection. The acquired satellites are the closest to the collection point at that time, which confirms that the data set is valid.

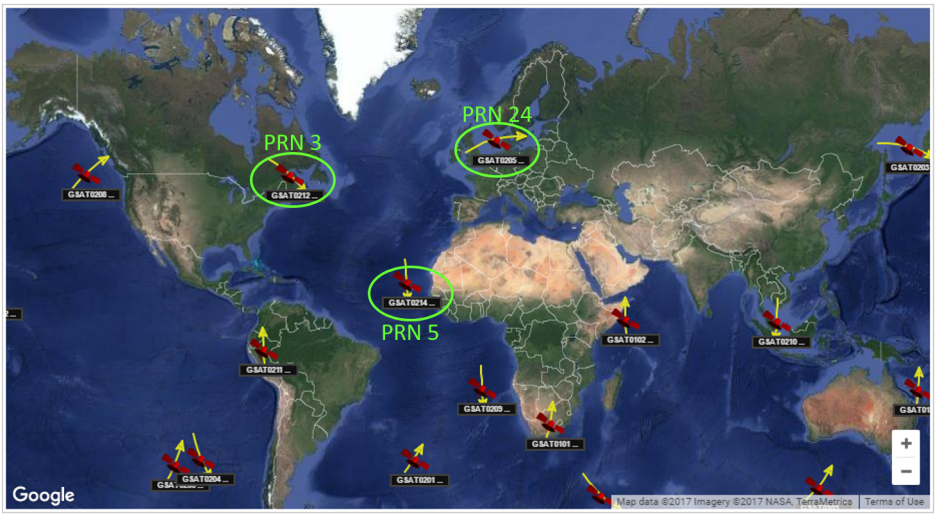
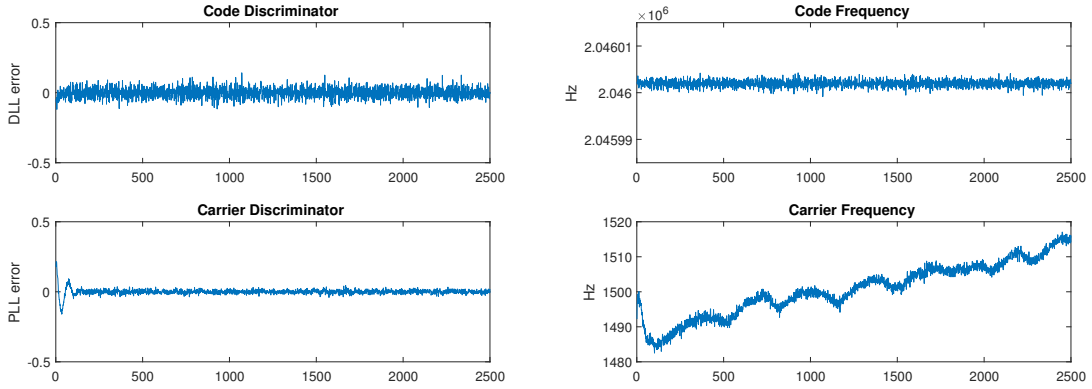


Figure 4.11: Live satellites at the moment of the data collection.

Satellite PRN 24 is analyzed. Figure 4.12(a) shows the code and carrier discriminators' outputs oscillating around zero and Figure 4.12(b) the code and carrier frequency, which seem to be stable. In this case, the carrier frequency is increasing over time, which means that the satellite is moving towards the receiver. This fact is also illustrated by a yellow arrow in Figure 4.11



(a) USRP signal discriminators' output (b) USRP signal code and carrier frequencies

Figure 4.12: USRP signal discriminators' output (a), code & carrier frequencies (b).

Finally, the prompt correlator's outputs are plotted in Figure 4.13 and Figure 4.14 shows a zoomed version. The energy is concentrated on the quadra-phase arm and the in-phase arm shows the symbols output.

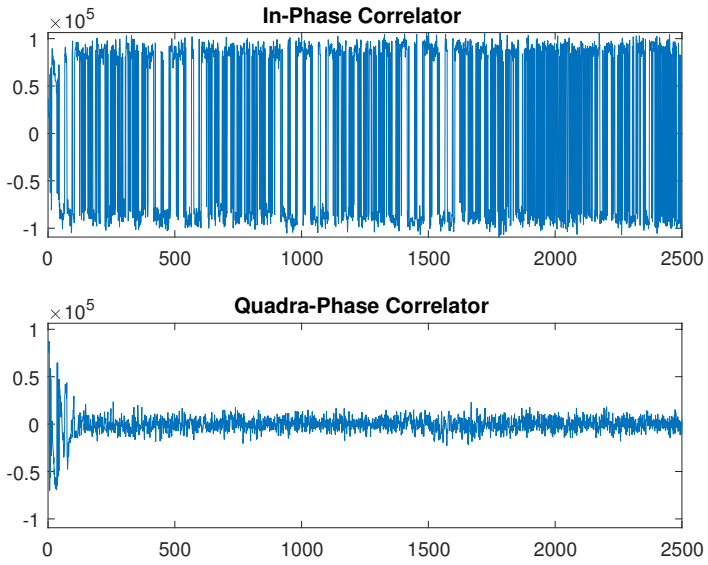


Figure 4.13: USRP signal prompt correlators' output.

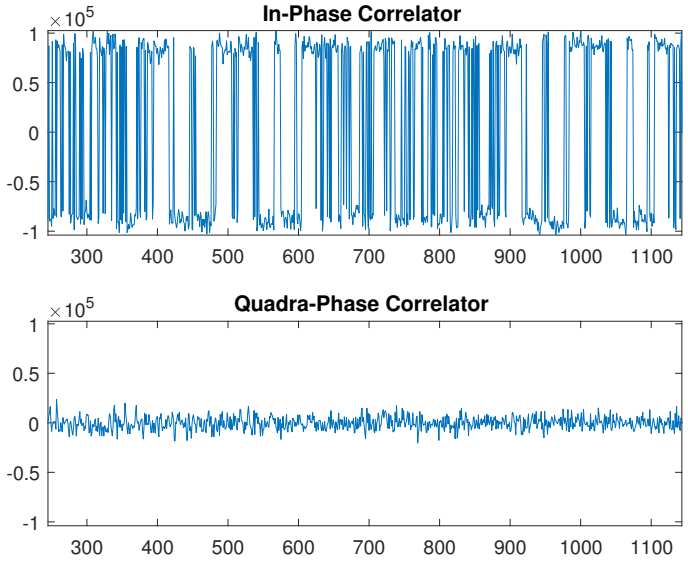


Figure 4.14: USRP signal prompt correlators' output zoomed.

In this chapter, the toolbox has been proved to be working with simulated signals, real signals recorded with a tested sampler and also signals collected with a USRP module. This way, this collection system has also been tested and will be used to perform a multipath analysis of the Galileo system.

CHAPTER 5

Multipath solution and implementation

In this chapter, multipath error affecting Galileo is studied. A general background is provided and some multipath mitigation strategies are mentioned. The implemented solution is explained (as changes from the main system, Chapter 3). Finally, some multipath test results are shown and discussed.

5.1 Background

The multipath concept refers to the error produced by the reception of reflected versions of the signal in addition to the direct path signal. [16] describes an arbitrary signal as:

$$A_D = A_0 \sin(\omega t + \phi) \tag{5.1}$$

Where A_0 refers to the amplitude, ω to the angular frequency and ϕ to the phase. The multipath versions of the signal travel different distances, which is translated in a phase change at the antenna position. Thus, if k_M is just a scaling factor and ϕ_M the phase from reflexion M, a multipath version of the signal can be written as:

$$A_M = k_M A_0 \sin(\omega t + \phi + \phi_M) \tag{5.2}$$

At the antenna position, both the direct path signal and all the multipath signals are received, resulting in:

$$A = A_D + \sum_{i=1}^{N} A_{M_i} = A_0 \left(\sum_{i=0}^{N} [k_{M_i} \sin(\omega t + \phi + \phi_{Mi})] \right) \tag{5.3}$$

5.1.1 Multipath in positioning systems

Multipath error is among all the possible GNSS errors, the one that probably most affect the baseband processing (acquisition and tracking). As described in Section 2.3, the GNSS signal is RHCP, meaning that one reflection would change the polarization and the antenna would not receive the reflected signal. Therefore, the multipath efect in Galileo is visible after two reflecions (also visible if antenna is not properly RHCP).

Figure 5.1 shows a simple scheme describing a multipath environment example. [16] states: "In general, multipath affects signal power, code delays, carrier phases, Doppler, and signal scintillation parameters", thus it is important to mitigate it.

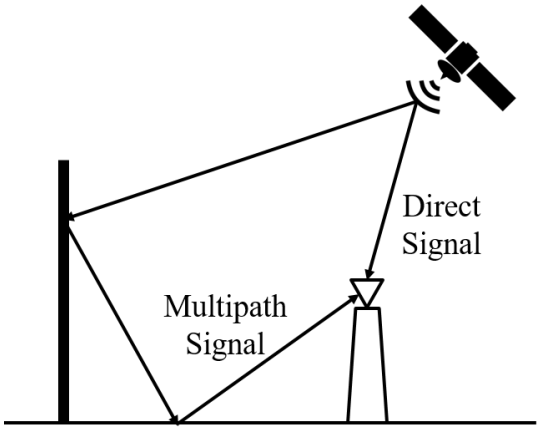


Figure 5.1: Multipath environment simple drawing.

Multipath distortion is often impacting the ACF shape. This effect is illustrated in Figure 5.2, where the blue line represents the reference ACF and the red line the one affected by multipath.

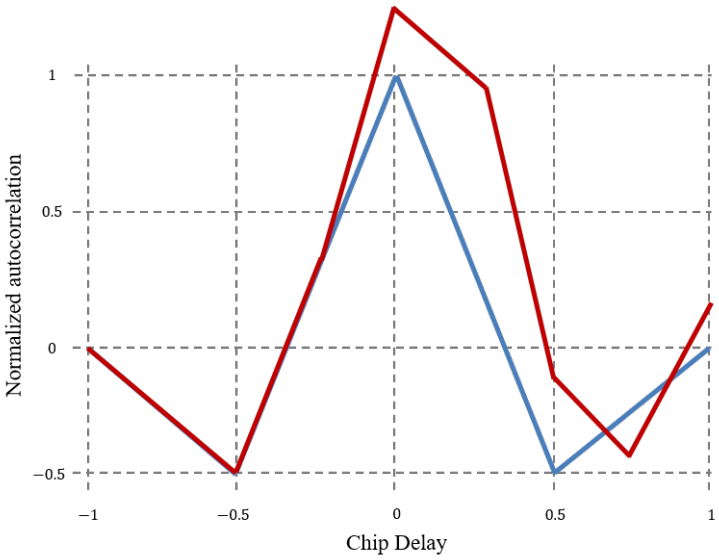


Figure 5.2: Multipath affected ACF.

It seems clear that changes in the correlation function will definitely impact the code tracking loop, since the early and late replicas will be distorted and the DLL will lock the code frequency at a wrong value. In the next section, some multipath mitigation strategies are introduced.

5.1.2 Multipath mitigation strategies

Since multipath is a very common phenomenon in any type of wireless communication, many different solutions have been proposed. For GNSS applications, these are the most extended:

1. Low-multipath antennas: GNSS antennas are often designed to mitigate multipath. [17] or [18] are good examples of low-multipath antennas.

2. Multipath estimation: proposed in [19], based on estimating the multipath effect inside the code tracking loop.

3. Multi-correlator receiver: the implemented solution for this project, based on a fine regeneration of the autocorrelation function. Many articles introduce the *very early* and *very late* correlators concept [20], [21] for unambiguous Galileo code tracking and a multi-correlator implementation will just extend this concept.

4. Other strategies: many other solutions: digital beamforming [22], enhanced frequency tracking [23], etc.

5.2 Implementation

The implemented strategy will be a modification of the original toolbox in order to accept a larger number of early and late correlators. This way, the ACF can be regenerated to mitigate multipath.

5.2.1 Multi-correlator receiver

The structure of the system is still the same as described in Chapter 3. The main blocks of the system described before in Figure 3.10 remain in the same order and only small changes are introduced in the tracking algorithm. At a glance, the main blocks' modifications are:

- Settings file: a new settings parameter is introduced: the number of early and late correlators. Initially this is set at 10 early and 10 late correlators.

- Main file: the only change is the way the tracking block is called, since it has different names for the regular Early Prompt Late (E-P-L) implementation and the Multi-Correlator (M-Cr) implementation.

- Acquisition: remains the same, since only the tracking is affected.

- Tracking & Results: there is a new tracking block based on the E-P-L function and new results to plot. This blocks will be described in detail later.

Figure 5.3 shows the new tracking block diagram with more than one early and late correlator. Notice the fact that the frequency tracking loop is not affected, since only the prompt samples are used. Therefore, only two main tasks are required: implementing the new correlators and changing the DLL discriminator.

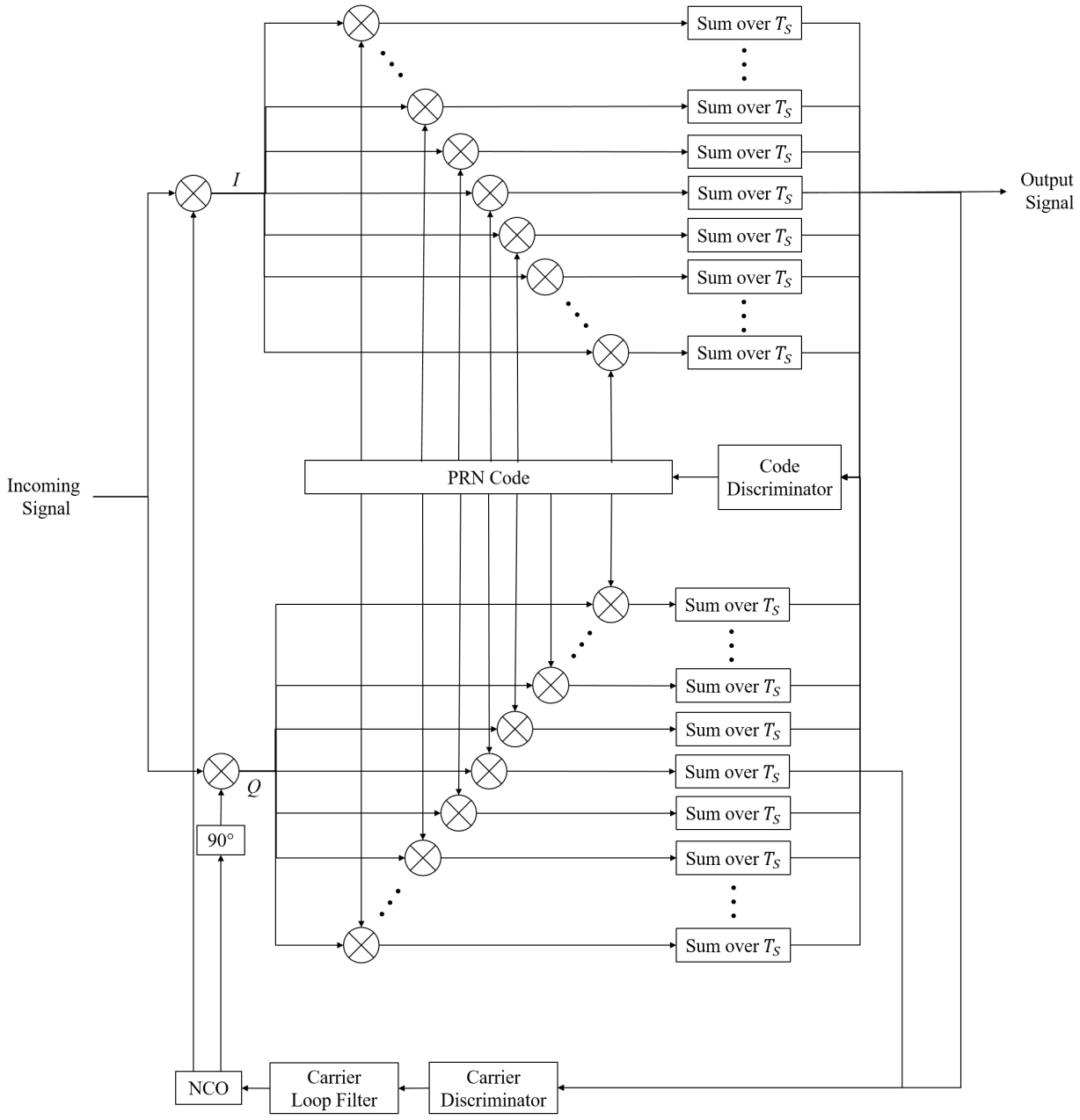


Figure 5.3: Multi-correlator tracking block diagram.

The tracking algorithm is similar to the one described in Algorithm 6. There are only three modifications: correlators' spacing, correlation function and DLL discriminator.

Correlators' spacing

Somehow from the new parameter (number of correlators), a spacing vector has to be generated. In this case, equally spaced correlators are implemented with spacing from -1 to 1 chip in order to regenerate the full autocorrelation function. The ideal sampling times are shown in Figure 5.4.

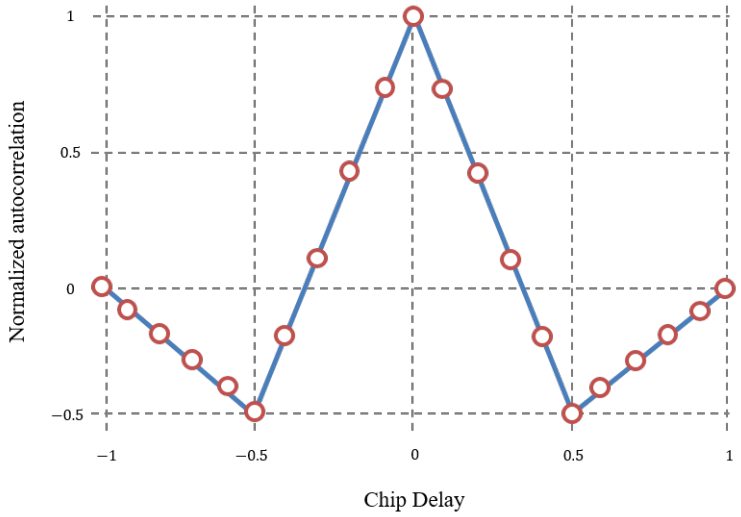


Figure 5.4: ACF spacing in the ideal case.

Correlation function

Similar to the one presented in Algorithm 7. Instead of having three outputs for the early and late arms respectively, now it stores the I and Q early and late samples in two vectors with the length of the correlators' spacing vector.

DLL discriminator

The original discriminator (equation (3.2)) is modified to cope with all the correlators' output. The new discriminator is:

$$\begin{aligned} I &= [I_{E_N}, ...I_{E_1}, I_P, I_{L_1}...I_{L_N}] \\ Q &= [Q_{E_N}, ...Q_{E_1}, Q_P, Q_{L_1}...Q_{L_N}] \end{aligned} \tag{5.4}$$

$$D_{DLL} = \frac{\sqrt{\sum_{i=1}^{N} I_{E_i}^2 + \sum_{i=1}^{N} Q_{E_i}^2} - \sqrt{\sum_{i=1}^{N} I_{L_i}^2 + \sum_{i=1}^{N} Q_{L_i}^2}}{\sqrt{\sum_{i=1}^{N} I_{E_i}^2 + \sum_{i=1}^{N} Q_{E_i}^2} + \sqrt{\sum_{i=1}^{N} I_{L_i}^2 + \sum_{i=1}^{N} Q_{L_i}^2}} \tag{5.5}$$

5.2.2 Autocorrelation function

Besides improving the performance to mitigate multipath, this implementation allows reconstruction of the autocorrelation function, since 21 samples are taken in every integration period (4 ms). Figure 5.5 shows the result: the ACF function over time when the satellite has already been tracked. Notice that the negative prompt samples have been flipped in order to illustrate the shape.

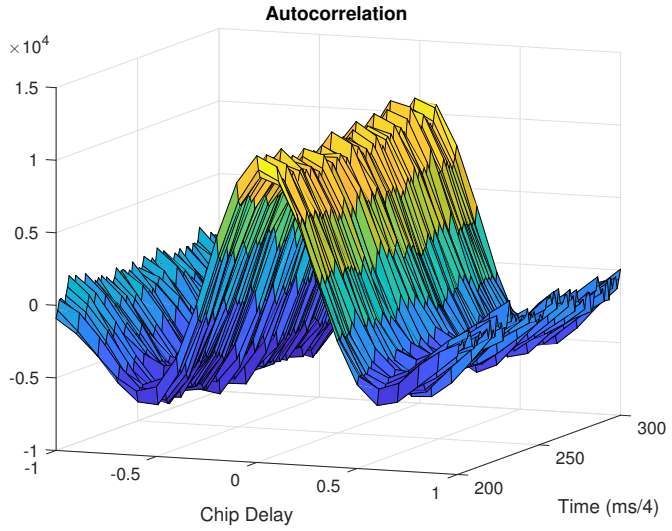


Figure 5.5: Recreation of the Autocorrelation function over time.

Figure 5.6 shows the first 100 integration periods. It is interesting to see the way the algorithm adjusts to recover the characteristic shape of the Galileo ACF.

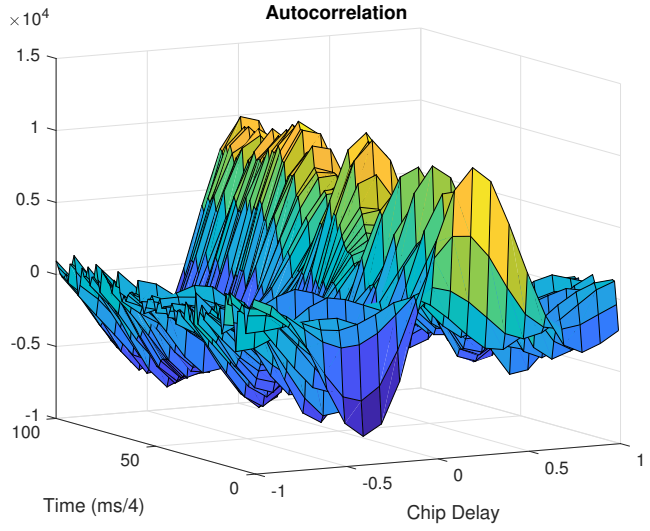


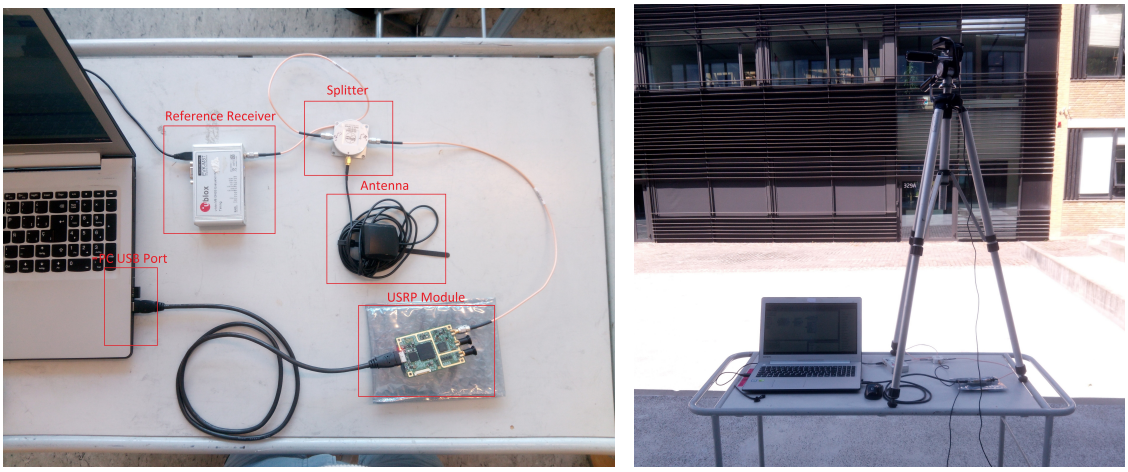
Figure 5.6: Recreation of the ACF over time during the transient time.

5.3 Results and performance

In this section, two signals collected simultaneously are studied. One of them is received by the roof antenna (multipath free) and the other is collected by a different antenna at place where multipath should severely impact the signal quality (DTU Space backyard).

5.3.1 Multipath environment set-up for data collection

Figure 5.7(a) and Figure 5.7(b) show the circuit and outside set-up respectively. A reference receiver is used to provide the bias current to the antenna using a splitter. Then the antenna is connected to the USRP module and this is connected to the PC. The antenna is placed on a tripod in order to place it above our heads.



(a) Circuit set-up (b) Outside set-up

Figure 5.7: Multipath data collection set-up.

Both signals are going to be processed by the toolbox. In order to easily understand the discussion, the multipath free signal will be referred to as *reference data* and the multipath affected signal as *multipath data*. Besides this, we will refer to the normal toolbox as the *E-P-L solution* and to the multi-correlator toolbox as *M-Cr*. Thus we will be looking at the results of the following tests:

- Reference data processed by E-P-L toolbox.
- Reference data processed by M-Cr toolbox.
- Multipath data processed by E-P-L toolbox.
- Multipath data processed by M-Cr toolbox.

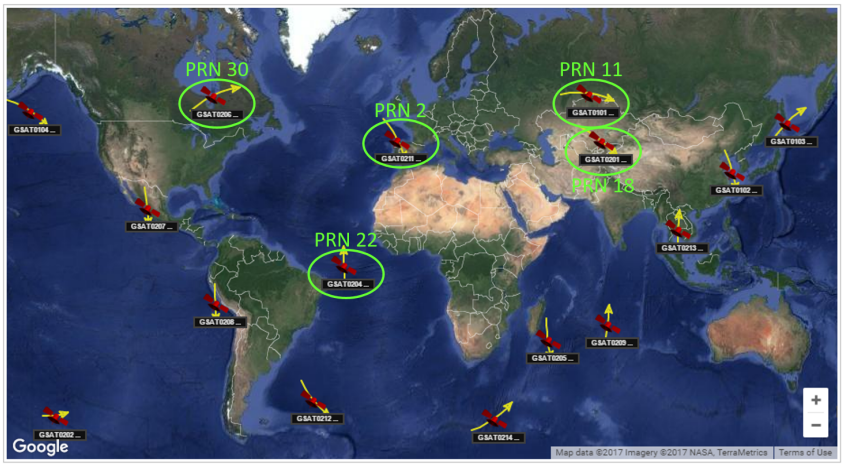
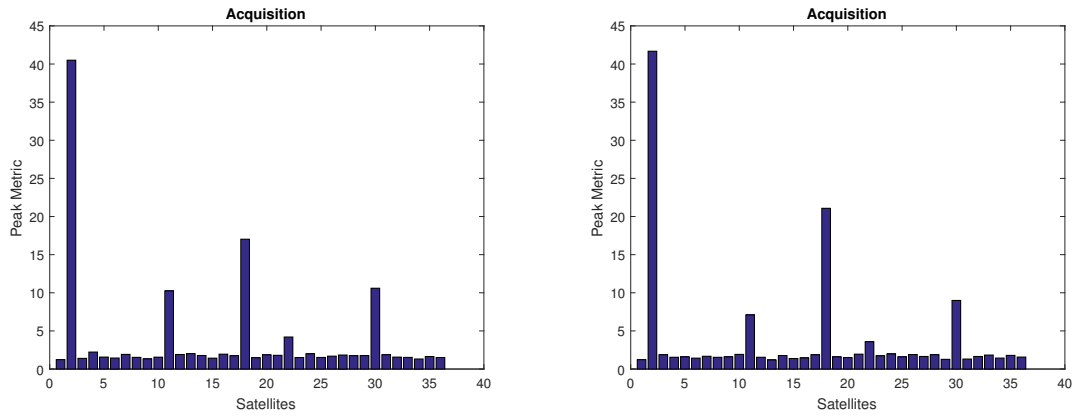


Figure 5.8: Live satellites at the moment of the data collection for the multipath test.

Figure 5.8 shows the live satellites at the moment of data collection. Figure 5.9(a) and Figure 5.9(b) show the acquisiton plots for the reference and the multipath signals respectively. Five satellites are acquired and they correspond to the ones marked in green color in the previous image. Note that the peak metric for both signals are similar. This happens because the gain for the outdoor signal has been increased 15 dB (to 45 dB).



(a) Reference data (b) Multipath data

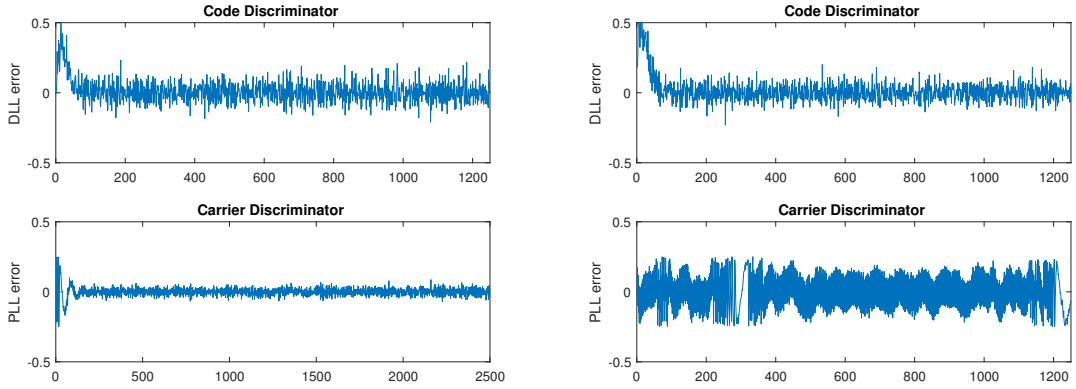
Figure 5.9: Acquisition results for the reference (a) and multipath (b) signals.

In the next sections, two comparisons are carried out:

1. Reference data vs. Multipath data processed by E-P-L toolbox to study the degradation of the tracking due to multipath.
2. Reference data vs. Multipath data processed by M-Cr toolbox to study the impact of the multi-correlator solution to improve tracking.

5.3.2 Multipath degradation of the collected signal

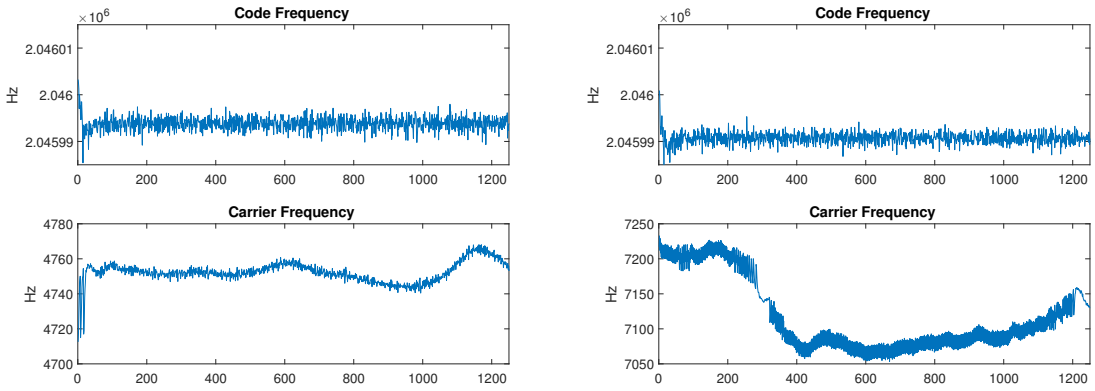
In this section, reference data and multipath data processed by E-P-L toolbox are studied. Plots correspond to satellite PRN 18. Figure 5.10(a) and Figure 5.10(b) show the code and carrier discriminators for both signals. For the multipath signal, the error seems to be larger and this should be translated in higher fluctiations of the carrier and code frequencies.



(a) Reference data (b) Multipath data

Figure 5.10: Discriminators of the reference and multipath signals with E-P-L solution.

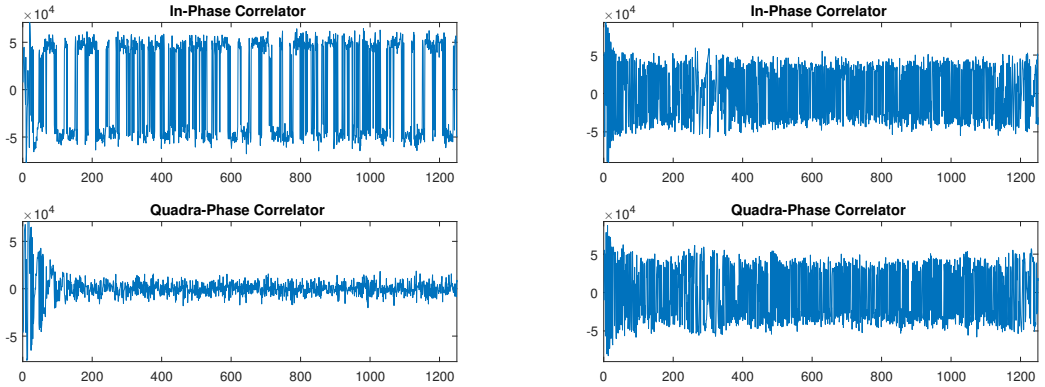
Figure 5.11(a) and Figure 5.11(b) illustrate the fact mentioned above about the carrier frequency, which has a range of 20 Hz for the reference signal and 200 Hz for the multipath signal. This is expected to be impacting the tracking loop and will definitely affect the output data quality.



(a) Reference data (b) Multipath data

Figure 5.11: Frequencies of the reference and multipath signals with E-P-L solution.

For a final comparison, the correlators' outputs are plotted in Figure 5.12(a) and Figure 5.12(b). In this case it is very clear that the multipath signal degradates the tracking performance, since the quadra-phase correlator and the in-phase correlator almost have the same energy and the information can not be seen.

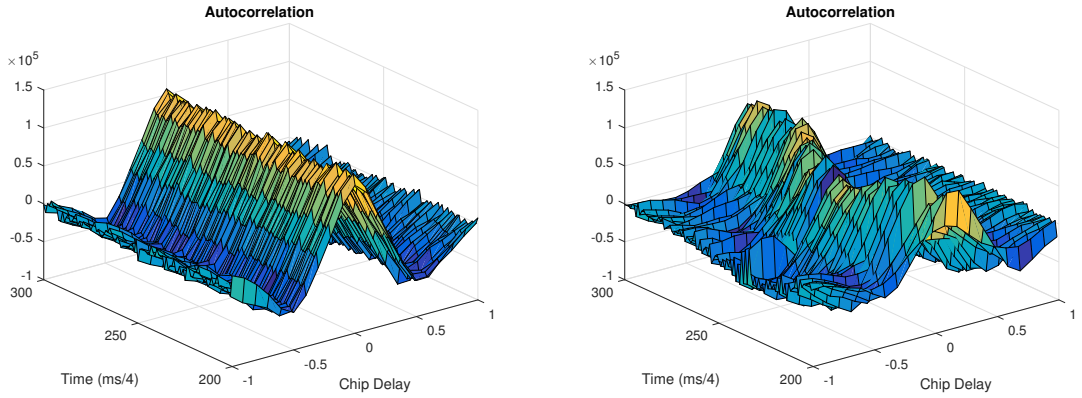


(a) Reference data (b) Multipath data

Figure 5.12: Correlators of the reference and multipath signals with E-P-L solution.

5.3.3 Multi-correlator solution impact on the performance

In this section, reference data and multipath data processed by M-Cr toolbox are studied. An improvement with respect to the previous results is expected in both signal sets, but it should have more impact in the multipath signal.

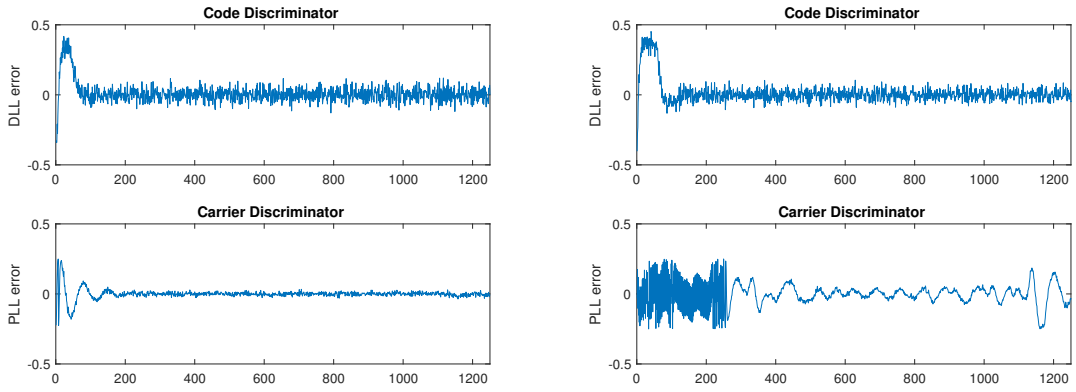


(a) Reference data (b) Multipath data

Figure 5.13: ACF over time for the reference and multipath signals with M-Cr solution.

Figure 5.13(a) and Figure 5.13(b) show the ACF plots for the reference and the multipath signal respectively. It seems clear that the second one has been degraded but the system is still able to reconstruct the autocorrelation.

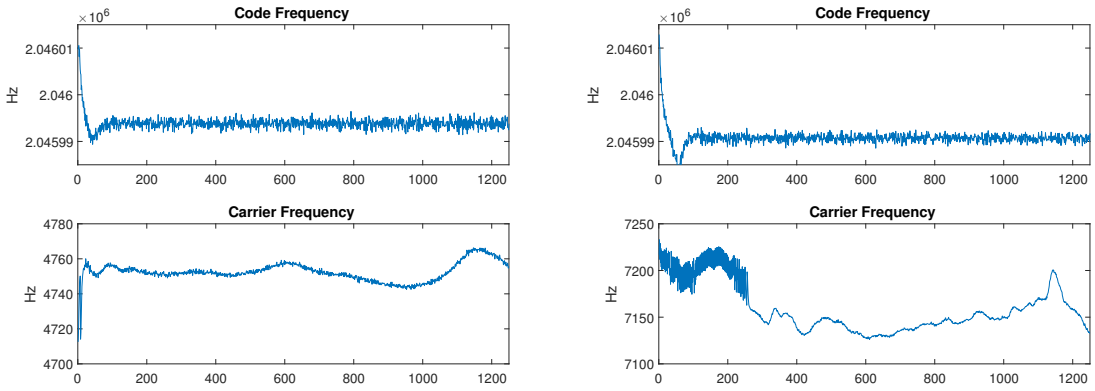
The discriminators' output are plotted in Figure 5.14(a) and Figure 5.14(b). There is a significant improvement in comparison with Figure 5.10(a) and Figure 5.10(b), for both signals and both carrier and code errors.



(a) Reference data (b) Multipath data

Figure 5.14: Discriminators of the reference and multipath signals with M-Cr solution.

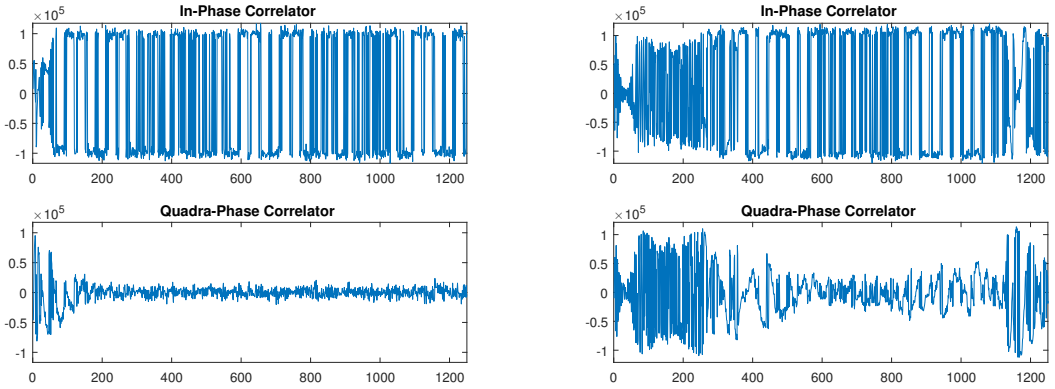
Figure 5.15(a) and Figure 5.15(a) illustrate the improvement in carrier and code frequency, as now the carrier frequency of the multipath signal ranges 60 Hz (vs. 200 Hz with the E-P-L solution).



(a) Reference data (b) Multipath data

Figure 5.15: Frequencies of the reference and multipath signals with M-Cr solution.

Again to support the results, the output of the correlators is also shown (Figure 5.16(a) and Figure 5.16(b)). The amplitude of the output signal has increased and the quadra-phase energy has been reduced. On top of this, information symbols is now visible.



(a) Reference data

(b) Multipath data

Figure 5.16: Correlators of the reference and multipath signals with M-Cr solution.

There is something in the results that may look extrange: the carrier frequencies from the same satellite for the reference and mltipath signal seem to be different. This effect is taken from the acquisition function, which outputs two different values for the signals. Taking a look to the other satellites' carrier frequencies, a constant offset of 2.4 kHz is observed, as pointed in Table 5.1 for satellits with PRN 11 and PRN 18, where the frequency differences are 2.422 kHz and 2.460 kHz respectively.

Table 5.1: Carrier frequency offset for the reference and multipath signals.

Frequency/Satellite	PRN 11	PRN 18
Reference Data	457,76	4734,99
Multipath Data	2880,09	7195,47
Frequency Offset	2422,33	2460,48

It is important to notice and clarify that both signals' outputs have improved. Therefore, the toolbox performance is increased not only for multipath affected signals but also for multipath free signals. However, extra computational resources are necessary, and perhaps for high quality signals, the performance of the E-P-L toolbox is enough.

5.4 Galileo vs. GPS

The multi-correlator implementation has been slightly modified to be working with GPS signals as well in order to be able to compare the performance between these two systems.

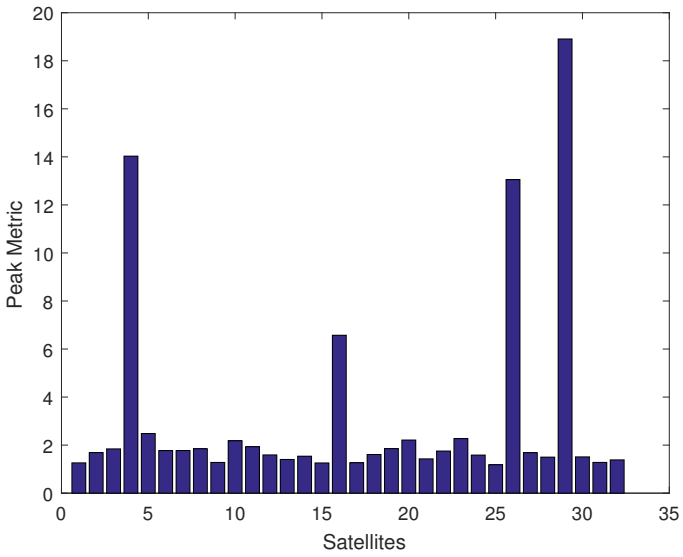


Figure 5.17: GPS signal acquisition plot.

Figure 5.17 shows the acquisition plot with four acquired satellites. PRN 29 has been analyzed. Figure 5.18 recreates the GPS ACF. Some multipath is appreciated in the shape, changing over time, increasing and decreasing in amplitude. Also, some changes in the traditional BPSK triangular shape illustrated in Figure 2.7 are noticed, which was mentioned before as the main multipath degradation effect.

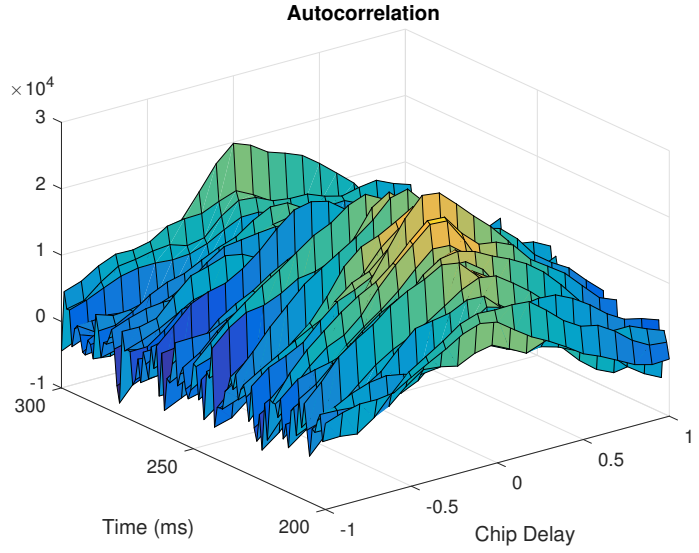
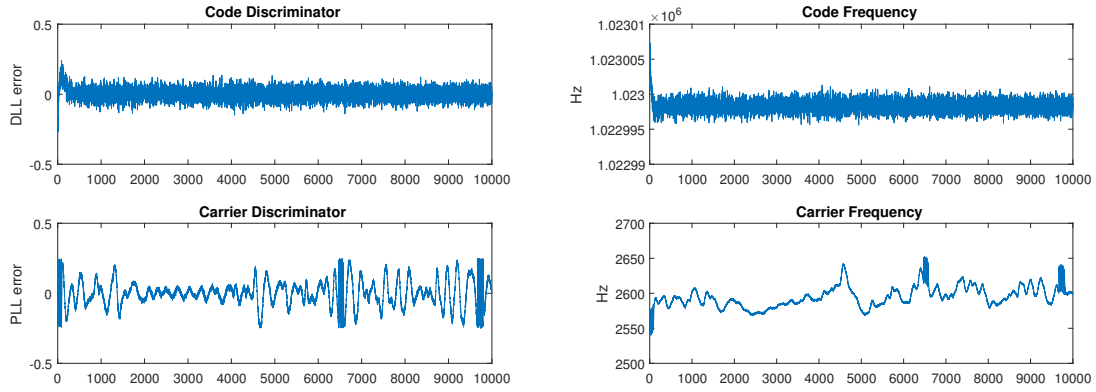


Figure 5.18: Recreation of the ACF over time during of a GPS signal.

Figure 5.19(a) and Figure 5.19(b) show the GPS discriminators' output and code and carrier frequencies respectively. In the GPS case, the results are similar to Galileo but the frequency range is a bit higher.



(a) GPS signal discriminators' output

(b) GPS signal code and carrier frequencies

Figure 5.19: GPS signal discriminators' output (a), code & carrier frequencies (b).

Finally, Figure 5.20 plots the prompt correlators' output. The performance is similar to the Galileo case, with some of the energy still in the quadra-phase arm.

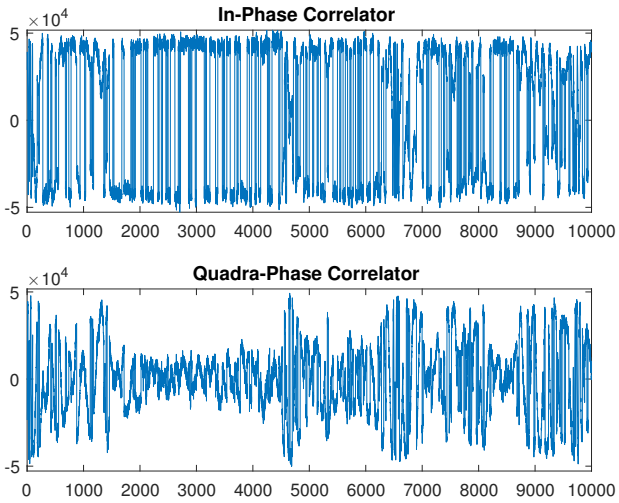


Figure 5.20: GPS signal prompt correlators' output using the multi-correlator implementation.

CHAPTER 6

Relation to UAVs

The DTU Space department is very interested in working with the so called Unmanned Aerial Vehicles (UAV), often called drones (see for example [8]). One of the purposes of using this UAVs is to fly over water to measure water level.

First, the use of SDR receivers is convenient since they are more lightweight than hardware receivers and often cheaper and less power consuming, which make it ideal for UAVs.

Second, multipath over water is a very relevant effect to consider. When flying over water, the GNSS antenna receives not only the Line Of Sight (LOS) direct signal, but also one or more multipath signals reflected in the water surface. Figure 6.1 illustrates this fact.

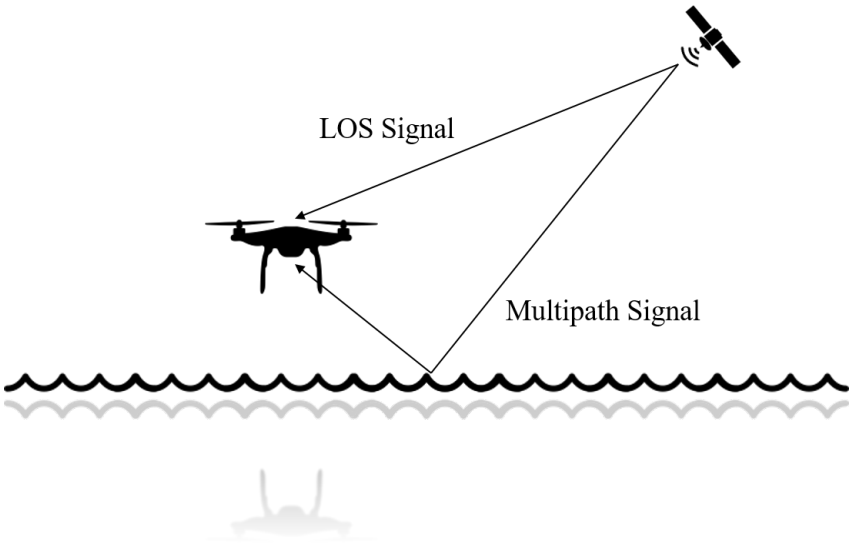


Figure 6.1: Typicall scenario for water level measurement with UAVs.

This multipath signal reaches the antenna later than the direct one, and as explained in Chapter 5, could be a source of destructive interference. The higher the drone flies, the easier is to separate the two signals.

In this case, the toolbox developed for this project is interesting and relevant due to two main reasons:

1. The evident improvement refers to the multi-correlator implementation described in Chapter 5 to mitigate multipath, which would be interesting to test in flying over water conditions.

2. The underlying reason is the Galileo autocorrelation function. Described in Section 2.2, Galileo is using a BOC modulation and GPS a BPSK modulation. Figure 2.7 shows the ACF functions for both signals, where it seems clear that the modulation with the narrower ACF (Galileo) will be more robust against multipath.

Thus, the fact that the toolbox is a part of a SDR receiver and that Galileo is the positioning system being treated, together with the multipath implemented solution, make this topic very interesting for UAVs.

CHAPTER 7

Conclusion

As introduced in Chapter 1, Galileo has been designed to improve the performance of the other satellite positioning systems (GPS, GLONASS, etc.). The constellation will be complete and fully working within the next years, which together with the achievements in high speed digital signal processing and the SDR boom, make this project of a great interest.

After this project, the toolbox has been adapted to perform Galileo acquisition and tracking and has been tested and checked with simulated signals, externally collected signals and even signals collected during the project with a USRP module. Besides this, in this thesis some background about satellite positioning, the Galileo signal and receiver operations has been presented.

Furthermore, a multipath case study has also been presented, including some background, an alternative toolbox with some modifications to mitigate this effect and a results and performance section. This implementation has also worked satisfactorily and it is of a great relevance for the finally described use in UAVs of the toolbox.

7.1 Future work

Towards the Global Positioning (GNSS), the focus should not be only into GPS or Galileo, but on both. The toolbox was initially developed by DTU Space for GPS and has now been adapted to Galileo; however, it is extremely important to merge both into a single baseband processing block.

Furthermore, all the position processing from pseudorange calculations to position calculation is out of the scope of this thesis and will have to be implemented in a future time.

Finally, the initial concern was and still is to use the toolbox on a UAVs (see Chapter 6), which indeed requires a different hardware to run the toolbox (a PC has been used so far) and probably some improvements in the implementation in terms of memory usage, computational resources, real-time processing, etc.

Nevertheless, the results and performance can be rated as very good, as well as the general impression of the toolbox and the multipath mitigation implementation.

APPENDIX A

GPS signal characteristics

In this appendix, some general ideas are provided regarding GPS signal. It is important to know at least the basic concepts since this system has been working for more than two decades and many Galileo specifications are based on GPS features. Therefore, the characteristics described in Section 2.2 can be seen as improvements of the GPS initial features.

A.1 Frequency bands and modulation

Figure 2.5 already shows the GPS bands together with the Galileo bands. In this project, only the E1 band is studied, which in GPS corresponds to the L1 band.

For this L1 frequency band, the modulation used by the GPS system is BPSK. This modulation does not use any subcarrier, thus only one peak centered at the carrier frequency will appear (two peaks appear in the Galileo spectrum, Figure 2.9).

Furthermore, the autocorrelation function is just a triangle function centered in the relative time 0 and the peak value is 1. The autocorrelation function is plotted together with the Galileo one in Figure 2.7.

Finally, in order to show a demonstration of improvement, Figure A.1 illustrates the signal generation process for GPS. In a similar way, Figure 2.6 adds the subcarrier to the same signal and the final transmitted signal changes.

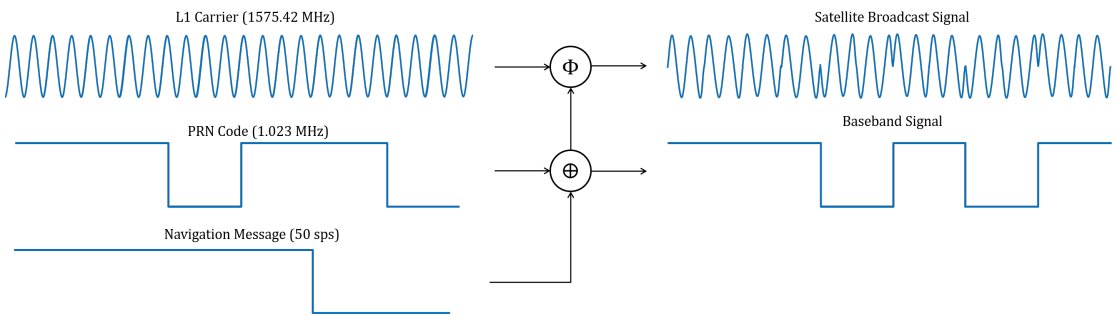


Figure A.1: GPS signal generation process.

A.2 Signal generation

According to Figure A.1, GPS signal generation is an easy process. Figure A.2 shows the block diagram, consisting only in one block which multiplies the navigation message and the C/A PRN code.

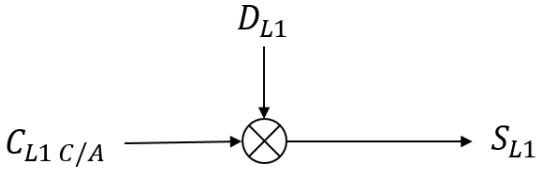


Figure A.2: Block diagram for the L1 signal generation.

The signal is described in detail in (A.1), taking the generated signal and multiplying by the pulse.

$$S_{L1}(t) = \sum_{l=-\infty}^{+\infty} C_{L1C/A}\left(\mid l \mid_{1023}\right) \oplus D_{L1}\left(l\right) \cdot p(t - lT_{c,L1}) \tag{A.1}$$

It is important to notice that altough the code rate is the same as for Galileo (1,023 MHz), the duration in time and the length of the code are four times less (4 ms and 1023 chips respectively). This, together with the data rate (50 bps), means that for every symbol, 20 PRN codes are necessary, while for Galileo only one already contains the information of one symbol.

Finally, as a comparison tool, Table A.1 shows the main GPS L1 C/A signal characteristics and the differences that Galileo has introduced.

Table A.1: GPS L1 C/A signal characteristics and differences with Galileo E1.

Signal Component	L1 C/A	Galileo differences
Carrier Frequency	1575,42 MHz	1575,42 MHz
Access Technique	CDMA	CDMA
Modulation	**BPSK**	**CBOC (6,1,1/11)**
Subcarrier Frequency	–	**1,023 MHz**
Code Frequency	1,023 MHz	1,023 MHz
Code Length	**1023**	**4092**
Code Time Length	**1 ms**	**4 ms**
Data Rate	**50 sps**	**250 sps**

APPENDIX B

USRP B200 mini

The chosen device to collect IF samples is a Universal Software Radio Peripheral – USRP, designed and sold by *Ettus Research (National Instruments)*. According to the manufacturer,"the USRP family of products is designed for RF applications from DC to 6 GHz", which includes the GNSS frequency band.

Precisely, the model used is the USRP B200 mini (shown in Figure B.1), that includes a *AD9364* converter from *Analog Devices* and a *Spartan-6* FPGA by *Xilinx*. Among others, the B200 mini offers the following capabilities:

- Integrated RF frontend (70 MHz - 6 GHz)
- External 10 MHz reference input
- Configurable clock rate
- Variable analog bandwidth (200 kHz - 56 MHz)

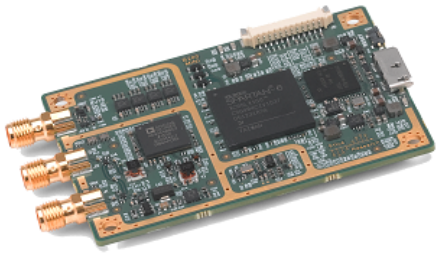


Figure B.1: USRP B200 mini. Source: Ettus Research.

The working priciple is simple: the board can work in transmitting or receiving mode (two different antenna ports and a third one for the external reference signal). A Universal Serial Bus (USB) port connects the USRP module with a PC that would act as the data source (transmitting mode) or the data sink (receiving mode). A block diagram from *Ettus Research* showing this working principle is provided in Figure B.2

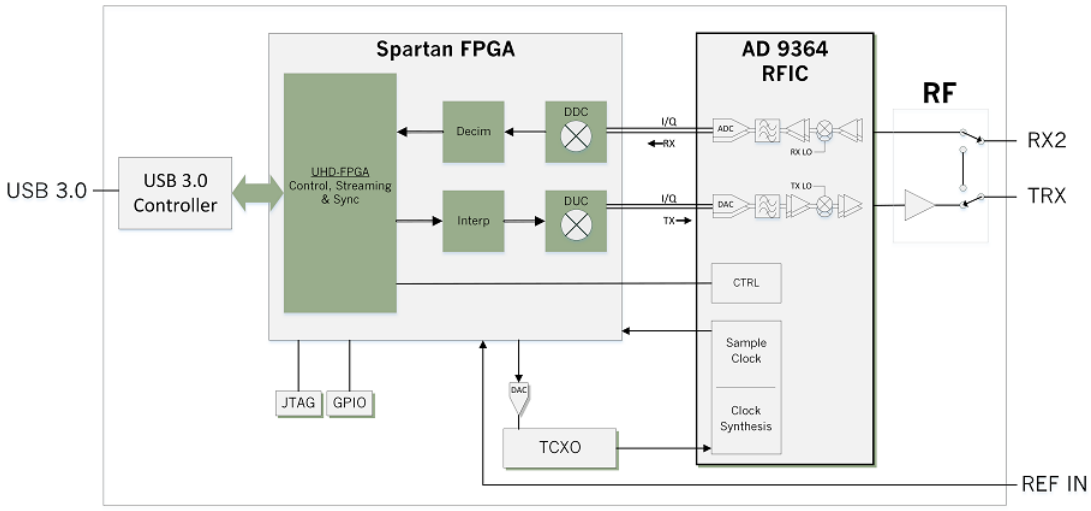


Figure B.2: USRP B200 mini architecture. Source: Ettus Research.

The company has developed a GNU Radio software code repository to be able to use the board together with GNU Radio (see Appendix C).

APPENDIX C

GNUradio Companion

According to the GNU Radio web page, "GNU Radio is a free & open-source software development toolkit that provides signal processing blocks to implement software radios". The GNU Radio applications are known as *flowgraphs*.

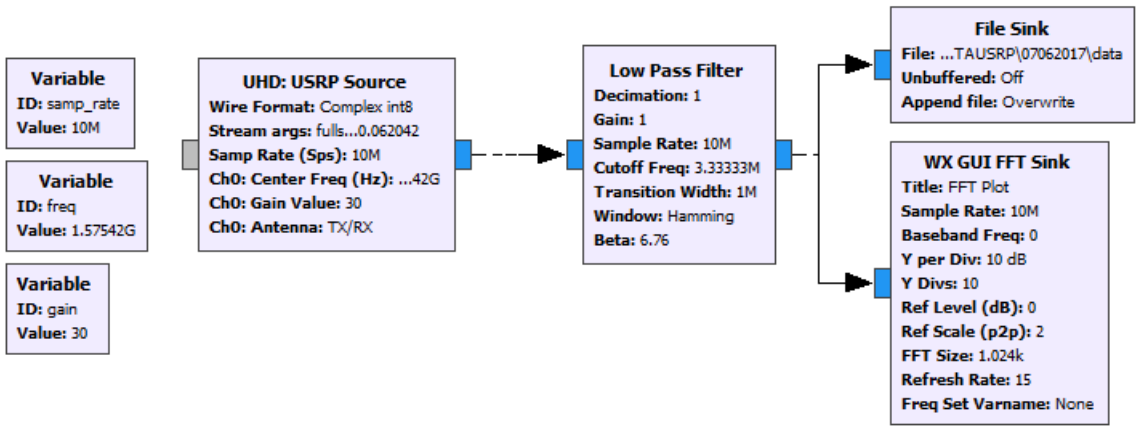


Figure C.1: GNU Radio flowchart for IF data collection.

For this project, a Graphical User Interface (GUI) called GNU Radio Companion has been used. This is just a front end to the GNU Radio libraries for signal processing. Figure C.1 shows the flowgraph used for collecting IF samples. The main blocks are:

UHD source The USRP Hardware Driver (UHD) is just part of the code repository developed by *Ettus Research*. Every I and Q sample is stored as an 8-bit sample into a *float* data type. The sample rate is chosen to be 10 MHz and the gain 30 dB (this parameter will vary when collecting data outside).

Loww Pass Filter The idea is to filter the noise out of the band of interest. The cutoff frequency is 3.33 MHz.

File sink The purpose of this block is just to create or open a file in the PC and write the incoming data from the UHD block.

APPENDIX D

Data conversion

As mentioned in Chapter 4 (Section 4.1), data conversion is needed before the file can be processed by the toolbox. The script shown below is simple: it reads the input file, converts it to 8-bit samples and writes the output in the same path.

```
function dataConversion(fPath,file_in)
% dataConversion(path,file_in) reads the data contained in file_in,
% converts its content to 8-bit samples after scaling and writes the
% result to a file in the same path.

fid = fopen(strcat(fPath,file_in));  % Open file_in
a = fread(fid,'float')';             % Read the whole file
fclose(fid);                         % Close file

a = a*(128/max(abs(a)));             % Scale values to 8-bit maximum (128)
b = int8(a);                         % Convert data to 8-bit samples

file_out = strcat(file_in, '_out'); % Output file name (file_in + _out)

fidd = fopen(strcat(fPath,file_out),'w'); % Open or create output file
fwrite(fidd,b,'int8');               % Write data as 8-bit samples
fclose(fidd);                        % Close output file
end
```

.

Bibliography

[1] *Software Defined Radio*. John Wiley Sons, 2004 (cited on page 1).

[2] Carles Fernandez-Prades et al. "An Open Source Galileo E1 Software Receiver". In: *IEEE-ESA Workshop on Satellite Navigation Technologies and European Workshop on GNSS Signals and Signal Processing* (2012), page 6423057. DOI: 10.1109/NAVITEC.2012.6423057 (cited on pages 1, 3).

[3] R. Falone et al. "SDR GNSS receivers: A comparative overview of different approaches". In: *2014 IEEE Metrology for Aerospace (MetroAeroSpace)* (2014), pages 326–331. DOI: 10.1109/MetroAeroSpace.2014.6865943 (cited on page 1).

[4] L. L. Presti et al. "Software Defined Radio technology for GNSS receivers". In: *2014 IEEE Metrology for Aerospace (MetroAeroSpace)* (2014), pages 314–319. DOI: 10.1109/MetroAeroSpace.2014.6865941 (cited on page 1).

[5] "Satellite navigation: Galileo". In: *Summaries of EU legislation* (1999). URL: http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=LEGISSUM:l24205 (cited on page 2).

[6] Daniel Madelung Olesen, Jakob Jakobsen, and Per Knudsen. "Software-Defined GPS Receiver Implemented on the Parallella-16 Board". In: (2015), pages 3171–3177 (cited on pages 2, 17).

[7] European GNSS (Galileo) Open Service. *Signal In Space Interface Control Document*. European Union, 2016 (cited on page 3).

[8] Daniel Madelung Olesen et al. "GNSS Software Receiver for UAVs". In: *European Journal of Navigation* June (2016). Published in Special Issue of European Journal of Navigation (June 2016) (cited on pages 3, 51).

[9] Kai Borre et al. *A Software-Defined GPS and Galileo Receiver: a Single-frequency Approach*. 1st edition. Birkhäuser Verlag GmbH, 2007 (cited on pages 3, 11, 13, 16).

[10] Adam Lipka and Rafal Niski. "The concept of the GALILEO receiver". In: *IEEE Eurocon 2007: the International Conference on Computer As a Tool, Vols 1-6* (2007), pages 595–601. DOI: 10.1109/EURCON.2007.4400322 (cited on pages 3, 13).

[11] Elmar Wasle Bernhard Hofmann-Wellenhof Herbert Lichtenegger. *GNSS - Global Navigation Satellite Systems: GPS, GLONASS, Galileo, and more*. 1st edition. Springer, 2007 (cited on page 3).

[12] Pabitra Kumar Ray Bernard Sklar. *Digital Communications. Fundamentals and Applications*. 2nd edition. Pearson, 2014, pages 770–783 (cited on page 4).

[13] C.A. Balanis. *Antenna theory*. John Wiley, 2005 (cited on page 11).

[14] Y. (ed.) Sun. *Wireless communication circuits and systems*. The Institution of Electrical Engineers, 2004 (cited on page 11).

[15] Mark Petovello et al. "Are there low-cost and low-weight options for GNSS IF storage?" In: *Inside GNSS* 2016.September/October (2016), pages 40–45 (cited on page 12).

[16] Ivan G Petrovski; Toshiaki Tsujii. *Digital satellite navigation and geophysics : a practical guide with GNSS signal simulator and receiver laboratory*. 1st edition. Cambridge University Press, 2012, pages 184–195 (cited on pages 37, 38).

[17] Li Du and Yunqi Fu. "A Small Wideband Low-Multipath GNSS Antenna Using Resistive Film". In: *IEEE Antennas and Wireless Propagation Letters* 12 (2013), pages 1045–1048. DOI: 10.1109/LAWP.2013.2279159 (cited on page 39).

[18] Peng Yang et al. "A small low-multipath GNSS antenna using annular slot loaded ground plane". In: *2016 IEEE 5th Asia-pacific Conference on Antennas and Propagation (ApCAP). Proceedings* (2016), pages 349–50, 349–350. DOI: 10.1109/APCAP.2016.7843237 (cited on page 39).

[19] R.D.J. van Nee. "The Multipath Estimating Delay Lock Loop". In: *IEEE Second International Symposium on Spread Spectrum Techniques and Applications* (1992), pages 39, 40, 41, 42, 39–42. DOI: 10.1109/ISSSTA.1992.665623 (cited on page 39).

[20] Pavel Kovář et al. "Galileo Receiver Core Technologies". In: *Journal of Global Positioning Systems* 4.12 (2005), pages 176–183. DOI: 10.5081/jgps.4.1.176, 10.5081/jgps (cited on page 39).

[21] Adam Lipka and Rafal Niski. "The concept of the GALILEO receiver". In: *IEEE Eurocon 2007: the International Conference on Computer As a Tool, Vols 1-6* (2007), pages 595–601. DOI: 10.1109/EURCON.2007.4400322 (cited on page 39).

[22] Jose Lopez Vicario et al. "A Novel Look into Digital Beamforming Techniques for Multipath and Interference Mitigation in Galileo Ground Stations". In: *IEEE Advanced Satellite Multimedia Systems Conference and the ... Signal Processing for Space Communications Workshop* (2010), pages 240–247. DOI: 10.1109/ASMS-SPSC.2010.5586884 (cited on page 39).

[23] V. Dehghanian, J. Nielsen, and G. Lachapelle. "Enhanced GNSS Frequency Tracking in Multipath Environments". In: *IEEE Conference Proceedings - Canadian Conference on Electrical and Computer Engineering* (2013), pages 314–319. DOI: 10.1109/CCECE.2013.6567732 (cited on page 39).

This project is related to tracking of the Galileo E1 OS signal. The objective is to perform the baseband processing of Galileo signals using Matlab, thus the scope of the project is within digital processing of radio-frequency signals.

The implementation is based on an existing toolbox developed at the DTU Space department for GPS, which has been adapted to cope with the Galileo signal. Besides this, some data has been collected using a software receiver and the toolbox has been tested. Furthermore, a multipath study has been carried out using a multi-correlator strategy.

The thesis is divided into four main blocks. The first one introduces the Galileo signal as well as some software receiver theory as a background for the implementation, which is described right afterwards. Then, the data collection set-up is shown and some results are plotted, along with a discussion. Finally, multipath is analyzed in a separated chapter consisting of a small theory section, the implementation modifications and the results section.

DTU Space
National Space Institute
Technical University of Denmark

Elektrovej, building 327
DK - 2800 Kgs. Lyngby
Tel (+45) 4525 9500
Fax (+45) 4525 9575

www.space.dtu.dk

{2.7182818284}ϑωερτυθιοπσδφγηξκλ